

# Table of Contents

<b>1</b>	<b>Accounts-account-email-domain-aliases:GET</b>	<b>1</b>
<b>2</b>	<b>GET /accounts/{account}/email/{domain}/aliases</b>	<b>2</b>
	2.1 Contents	2
	2.2 Request	2
	2.3 Response	2
	2.4 Examples	2
	2.5 See also	3
<b>3</b>	<b>Accounts-account-email-domain-blackwhitelist-address:DELETE</b>	<b>4</b>
<b>4</b>	<b>DELETE /accounts/{account}/email/{domain}/blackwhitelist/{address}</b>	<b>5</b>
	4.1 Contents	5
	4.2 Request	5
	4.3 Response	5
	4.4 Examples	5
<b>5</b>	<b>Accounts-account-email-domain-blackwhitelist-address:GET</b>	<b>6</b>
<b>6</b>	<b>GET /accounts/{account}/email/{domain}/blackwhitelist/{address}</b>	<b>7</b>
	6.1 Contents	7
	6.2 Request	7
	6.3 Response	7
	6.4 Examples	7
<b>7</b>	<b>Accounts-account-email-domain-blackwhitelist:GET</b>	<b>9</b>
<b>8</b>	<b>GET /accounts/{account}/email/{domain}/blackwhitelist</b>	<b>10</b>
	8.1 Contents	10
	8.2 Request	10
	8.3 Response	10
	8.4 Examples	11
<b>9</b>	<b>Accounts-account-email-domain-blackwhitelist:PUT</b>	<b>13</b>
<b>10</b>	<b>PUT /accounts/{account}/email/{domain}/blackwhitelist</b>	<b>14</b>
	10.1 Contents	14
	10.2 Request	14
	10.3 Response	14
	10.4 Examples	14
<b>11</b>	<b>Accounts-account-email-domain-cleanmailplus:GET</b>	<b>16</b>
<b>12</b>	<b>GET accounts/{account}/email/{domain}/cleanmailplus</b>	<b>17</b>
	12.1 Contents	17
	12.2 Request	17
	12.3 Response	17
	12.4 Examples	18
<b>13</b>	<b>Accounts-account-email-domain-cleanmailplus:PUT</b>	<b>20</b>
<b>14</b>	<b>PUT accounts/{account}/email/{domain}/cleanmailplus</b>	<b>21</b>
	14.1 Contents	21
	14.2 Request	21
	14.3 Response	21
	14.4 Examples	21
<b>15</b>	<b>Accounts-account-email-domain-contacts-contact:DELETE</b>	<b>24</b>
<b>16</b>	<b>DELETE /accounts/{account}/email/{domain}/contacts/{contact}</b>	<b>25</b>
	16.1 Contents	25
	16.2 Request	25
	16.3 Response	25
	16.4 Examples	25
	16.5 See also	25
<b>17</b>	<b>Accounts-account-email-domain-contacts-contact:GET</b>	<b>26</b>
<b>18</b>	<b>GET /accounts/{account}/email/{domain}/contacts/{contact}</b>	<b>27</b>
	18.1 Contents	27
	18.2 Request	27
	18.3 Examples	28
	18.4 See also	29
<b>19</b>	<b>Accounts-account-email-domain-contacts-contact:PUT</b>	<b>30</b>
<b>20</b>	<b>PUT /accounts/{account}/email/{domain}/contacts/{contact}</b>	<b>31</b>
	20.1 Contents	31
	20.2 Request	31
	20.3 Response	32
	20.4 Examples	32
	20.5 See also	33

# Table of Contents

<b>21 Accounts-account-email-domain-contacts:GET</b> .....	<b>34</b>
<b>22 GET /accounts/{account}/email/{domain}/contacts?filter{FieldName}={FieldValue}&amp;filter{FieldName}={FieldValue}</b> .....	<b>35</b>
<b>23 GET /accounts/{account}/email/{domain}/contacts?search{FieldName}={FieldValue}&amp;search{FieldName}={FieldValue}</b> .....	<b>36</b>
23.1 Contents.....	36
23.2 Request.....	36
23.3 Examples.....	37
23.4 See also.....	39
<b>24 Accounts-account-email-domain-contacts:POST</b> .....	<b>40</b>
<b>25 POST /accounts/{account}/email/{domain}/contacts</b> .....	<b>41</b>
25.1 Contents.....	41
25.2 Request.....	41
25.3 Response.....	42
25.4 Examples.....	42
25.5 See also.....	43
<b>26 Accounts-account-email-domain-infostore:GET</b> .....	<b>44</b>
<b>27 GET /accounts/{account}/email/{domain}/infostore</b> .....	<b>45</b>
27.1 Contents.....	45
27.2 Request.....	45
27.3 Response.....	45
27.4 Examples.....	45
<b>28 Accounts-account-email-domain-infostore:PUT</b> .....	<b>47</b>
28.1 Contents.....	47
28.2 Request.....	47
28.3 Response.....	47
28.4 Examples.....	47
28.5 See also.....	48
<b>29 Accounts-account-email-domain-statistics-mailboxTypeCount:GET</b> .....	<b>49</b>
<b>30 GET /accounts/{account}/email/{domain}/statistics/mailboxTypeCount</b> .....	<b>50</b>
30.1 Contents.....	50
30.2 Request.....	50
30.3 Response.....	50
30.4 Examples.....	50
30.5 See also.....	51
<b>31 Accounts-account-email-domain-usernames-mailboxName-autoresponder:GET</b> .....	<b>52</b>
<b>32 GET accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder</b> .....	<b>53</b>
32.1 Contents.....	53
32.2 Request.....	53
32.3 Response.....	53
32.4 Examples.....	53
<b>33 Accounts-account-email-domain-usernames-mailboxName-autoresponder:PUT</b> .....	<b>55</b>
<b>34 PUT accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder</b> .....	<b>56</b>
34.1 Contents.....	56
34.2 Request.....	56
34.3 Response.....	56
34.4 Examples.....	56
<b>35 Accounts-account-email-domain-usernames-mailboxName-blackwhitelist-address:DELETE</b> .....	<b>58</b>
<b>36 DELETE /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}</b> .....	<b>59</b>
36.1 Contents.....	59
36.2 Request.....	59
36.3 Response.....	59
36.4 Examples.....	59
<b>37 Accounts-account-email-domain-usernames-mailboxName-blackwhitelist-address:GET</b> .....	<b>60</b>
<b>38 GET /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}</b> .....	<b>61</b>
38.1 Contents.....	61
38.2 Request.....	61
38.3 Response.....	61
38.4 Examples.....	61
<b>39 Accounts-account-email-domain-usernames-mailboxName-blackwhitelist:GET</b> .....	<b>63</b>
<b>40 GET /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist</b> .....	<b>64</b>
40.1 Contents.....	64
40.2 Request.....	64
40.3 Response.....	64
40.4 Examples.....	65

# Table of Contents

<b>41</b>	<b>Accounts-account-email-domain-usernames-mailboxName-blackwhitelist:PUT</b>	<b>67</b>
<b>42</b>	<b>PUT /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist</b>	<b>68</b>
	42.1 Contents	68
	42.2 Request	68
	42.3 Response	68
	42.4 Examples	68
<b>43</b>	<b>Accounts-account-email-domain-usernames-mailboxName-cleanmailplus:GET</b>	<b>70</b>
<b>44</b>	<b>GET accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus</b>	<b>71</b>
	44.1 Contents	71
	44.2 Request	71
	44.3 Response	71
	44.4 Examples	72
<b>45</b>	<b>Accounts-account-email-domain-usernames-mailboxName-cleanmailplus:PUT</b>	<b>74</b>
<b>46</b>	<b>PUT accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus</b>	<b>75</b>
	46.1 Contents	75
	46.2 Request	75
	46.3 Response	75
	46.4 Examples	75
<b>47</b>	<b>Accounts-account-email-domain-usernames-mailboxName-exchange-action:POST</b>	<b>78</b>
<b>48</b>	<b>POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/action</b>	<b>79</b>
	48.1 Contents	79
	48.2 Request	79
	48.3 Response	80
	48.4 Examples	80
	48.5 See also	80
<b>49</b>	<b>Accounts-account-email-domain-usernames-mailboxName-exchange-messagestats:GET</b>	<b>81</b>
<b>50</b>	<b>GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats</b>	<b>82</b>
	50.1 Contents	82
	50.2 Request	82
	50.3 Response	82
	50.4 Examples	83
	50.5 See also	83
<b>51</b>	<b>Accounts-account-email-domain-usernames-mailboxName-exchange:DELETE</b>	<b>84</b>
<b>52</b>	<b>DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange</b>	<b>85</b>
	52.1 Contents	85
	52.2 Request	85
	52.3 Response	85
	52.4 Examples	85
<b>53</b>	<b>Accounts-account-email-domain-usernames-mailboxName-exchange:GET</b>	<b>86</b>
<b>54</b>	<b>GET accounts/{account}/email/{domain}/usernames/{mailbox}/exchange</b>	<b>87</b>
	54.1 Contents	87
	54.2 Request	87
	54.3 Response	87
	54.4 Examples	90
<b>55</b>	<b>Accounts-account-email-domain-usernames-mailboxName-exchange:PUT</b>	<b>92</b>
<b>56</b>	<b>PUT accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange</b>	<b>93</b>
	56.1 Request	93
	56.2 Response	94
	56.3 Examples	94
<b>57</b>	<b>Accounts-account-email-domain-usernames-mailboxName-forwards-fwdmailbox:DELETE</b>	<b>97</b>
<b>58</b>	<b>DELETE accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/{fwd-mailbox}</b>	<b>98</b>
	58.1 Contents	98
	58.2 Request	98
	58.3 Response	98
	58.4 Examples	98
	58.5 See also	98
<b>59</b>	<b>Accounts-account-email-domain-usernames-mailboxName-forwards:GET</b>	<b>99</b>
<b>60</b>	<b>GET accounts/{account}/email/{domain}/usernames/{mailbox}/forwards</b>	<b>100</b>
	60.1 Contents	100
	60.2 Request	100
	60.3 Response	100
	60.4 Examples	100
	60.5 See also	101

# Table of Contents

<b>61</b>	<b>Accounts-account-email-domain-usernames-mailboxName-forwards:POST</b>	<b>102</b>
<b>62</b>	<b>POST /accounts/{account}/email/{domain}/usernames/{mailbox}/forwards</b>	<b>103</b>
62.1	Contents	103
62.2	Request	103
62.3	Response	103
62.4	Examples	104
62.5	See also	106
<b>63</b>	<b>Accounts-account-email-domain-usernames-mailboxName-mailbox-action:POST</b>	<b>107</b>
<b>64</b>	<b>POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/action</b>	<b>108</b>
64.1	Contents	108
64.2	Request	108
64.3	Response	109
64.4	Examples	109
64.5	See also	109
<b>65</b>	<b>Accounts-account-email-domain-usernames-mailboxName-mailbox:DELETE</b>	<b>110</b>
<b>66</b>	<b>DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox</b>	<b>111</b>
66.1	Contents	111
66.2	Request	111
66.3	Response	111
66.4	Examples	111
<b>67</b>	<b>Accounts-account-email-domain-usernames-mailboxName-mailbox:GET</b>	<b>112</b>
<b>68</b>	<b>GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox</b>	<b>113</b>
68.1	Contents	113
68.2	Request	113
68.3	Response	113
68.4	Examples	114
<b>69</b>	<b>Accounts-account-email-domain-usernames-mailboxName-mailbox:PUT</b>	<b>115</b>
<b>70</b>	<b>PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox</b>	<b>116</b>
70.1	Contents	116
70.2	Request	116
70.3	Response	117
70.4	Examples	117
70.5	See also	119
<b>71</b>	<b>Accounts-account-email-domain-usernames-mailboxName-messagestats:GET</b>	<b>120</b>
<b>72</b>	<b>GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats</b>	<b>121</b>
72.1	Contents	121
72.2	Request	121
72.3	Response	121
72.4	Examples	122
72.5	See also	122
<b>73</b>	<b>Accounts-account-email-domain-usernames-mailboxName-migration:POST</b>	<b>123</b>
<b>74</b>	<b>POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/migration</b>	<b>124</b>
74.1	Contents	124
74.2	Request	124
74.3	Response	124
74.4	Examples	124
74.5	See also	125
<b>75</b>	<b>Accounts-account-email-domain-usernames-mailboxName-secrets:GET</b>	<b>126</b>
<b>76</b>	<b>GET /accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets</b>	<b>127</b>
76.1	Contents	127
76.2	Request	127
76.3	Response	127
76.4	Examples	127
76.5	See also	128
<b>77</b>	<b>Accounts-account-email-domain-usernames-mailboxName-secrets:PUT</b>	<b>129</b>
<b>78</b>	<b>PUT /accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets</b>	<b>130</b>
78.1	Contents	130
78.2	Request	130
78.3	Response	130
78.4	Examples	130
78.5	See also	131
<b>79</b>	<b>Accounts-account-email-domain-usernames-mailboxName-webmail:GET</b>	<b>132</b>
<b>80</b>	<b>GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail</b>	<b>133</b>
80.1	Contents	133
80.2	Request	133
80.3	Response	133

# Table of Contents

<b>80 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail</b>	<b>134</b>
80.4 Examples.....	134
<b>81 Accounts-account-email-domain-usernames-mailboxName-webmail:POST.....</b>	<b>135</b>
<b>82 POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail.....</b>	<b>136</b>
82.1 Contents.....	136
82.2 Request.....	136
82.3 Response.....	136
82.4 Examples.....	137
<b>83 Accounts-account-email-domain-usernames-mailboxName-webmail:PUT.....</b>	<b>138</b>
<b>84 PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail.....</b>	<b>139</b>
84.1 Contents.....	139
84.2 Request.....	139
84.3 Response.....	140
84.4 Examples.....	140
<b>85 Accounts-account-email-domain-usernames-mailboxName:GET.....</b>	<b>142</b>
<b>86 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}.....</b>	<b>143</b>
86.1 Contents.....	143
86.2 Request.....	143
86.3 Response.....	143
86.4 Examples.....	144
86.5 See also.....	144
<b>87 Accounts-account-email-domain-usernames:GET.....</b>	<b>145</b>
<b>88 GET /accounts/{account}/email/{domain}/usernames.....</b>	<b>146</b>
88.1 Contents.....	146
88.2 Request.....	146
88.3 Response.....	146
88.4 Examples.....	147
88.5 See also.....	148
<b>89 Accounts-account-email-domain-webmail:GET.....</b>	<b>149</b>
<b>90 GET /accounts/{account}/email/{domain}/webmail.....</b>	<b>150</b>
90.1 Contents.....	150
90.2 Request.....	150
90.3 Response.....	150
90.4 Examples.....	151
<b>91 Accounts-account-email-domain-webmail:PUT.....</b>	<b>152</b>
<b>92 PUT /accounts/{account}/email/{domain}/webmail.....</b>	<b>153</b>
92.1 Contents.....	153
92.2 Request.....	153
92.3 Response.....	153
92.4 Examples.....	153
<b>93 Accounts-account-email-domain-webmailLimits:GET.....</b>	<b>155</b>
<b>94 GET /accounts/{account}/email/{domain}/webmailLimits.....</b>	<b>156</b>
94.1 Contents.....	156
94.2 Request.....	156
94.3 Response.....	156
94.4 See also.....	156
<b>95 Accounts-account-email-domain-webmailNames:GET.....</b>	<b>157</b>
<b>96 GET /accounts/{account}/email/{domain}/webmailNames.....</b>	<b>158</b>
96.1 Contents.....	158
96.2 Request.....	158
96.3 Response.....	158
96.4 See also.....	158
<b>97 Accounts-account-email-domain-webmailUpgrade:GET.....</b>	<b>159</b>
<b>98 GET /accounts/{account}/email/{domain}/webmailUpgrade.....</b>	<b>160</b>
98.1 Contents.....	160
98.2 Request.....	160
98.3 Response.....	160
98.4 See also.....	160
<b>99 Accounts-account-email-domain:GET.....</b>	<b>161</b>
<b>100 GET /accounts/{account}/email/{domain}.....</b>	<b>162</b>
100.1 Contents.....	162
100.2 Request.....	162
100.3 Response.....	162
100.4 Examples.....	162

# Table of Contents

<b>101 Email API</b> .....	<b>164</b>
101.1 Mailboxes.....	164
101.2 Exchange.....	164
101.3 Aliases.....	164
101.4 Forwards.....	164
101.5 Autoresponder.....	164
101.6 CleanMailPlus.....	164
101.7 Webmail.....	164
101.8 Webmail Names.....	164
101.9 Black/White List.....	164
101.10 InfoStore.....	164
101.11 Mailbox domain statistics.....	164
101.12 Mailbox migration failure.....	164
101.13 Mailbox Message Count.....	164
101.14 Exchange Message Count.....	165
101.15 Email Domain.....	165
101.16 Domain Black/White List.....	165
101.17 Domain CleanMailPlus.....	165
101.18 Secret Questions.....	165
101.19 Secret Question Settings.....	165
101.20 Supported Secret Questions.....	165
101.21 Antivirus Antispam.....	165
<b>102 Email-domain-usernames-mailboxName-exchange-action:POST</b> .....	<b>166</b>
<b>103 POST /email/{domain}/usernames/{mailboxName}/exchange/action</b> .....	<b>167</b>
103.1 Contents.....	167
103.2 Request.....	167
103.3 Response.....	168
103.4 Examples.....	168
103.5 See also.....	168
<b>104 Email-domain-usernames-mailboxName-mailbox-action:POST</b> .....	<b>169</b>
<b>105 POST /email/{domain}/usernames/{mailboxName}/mailbox/action</b> .....	<b>170</b>
105.1 Contents.....	170
105.2 Request.....	170
105.3 Response.....	171
105.4 Examples.....	171
105.5 See also.....	171
<b>106 Email-domain-usernames-mailboxName-secrets:GET</b> .....	<b>172</b>
<b>107 GET /email/{domain}/usernames/{mailboxName}/secrets</b> .....	<b>173</b>
107.1 Contents.....	173
107.2 Request.....	173
107.3 Response.....	173
107.4 Examples.....	173
107.5 See also.....	174
<b>108 Email-domain-usernames-mailboxName-secrets:PUT</b> .....	<b>175</b>
<b>109 PUT /email/{domain}/usernames/{mailboxName}/secrets</b> .....	<b>176</b>
109.1 Contents.....	176
109.2 Request.....	176
109.3 Response.....	176
109.4 Examples.....	176
109.5 See also.....	177
<b>110 Email-domain-usernames-mailboxName-webmail:GET</b> .....	<b>178</b>
<b>111 GET /email/{domain}/usernames/{mailboxName}/webmail</b> .....	<b>179</b>
111.1 Contents.....	179
111.2 Request.....	179
111.3 Response.....	179
111.4 Examples.....	179
<b>112 Email-domain-usernames-mailboxName-webmail:POST</b> .....	<b>181</b>
<b>113 POST /email/{domain}/usernames/{mailboxName}/webmail</b> .....	<b>182</b>
113.1 Contents.....	182
113.2 Request.....	182
113.3 Response.....	182
113.4 Examples.....	183
<b>114 Email-domain-usernames-mailboxName-webmail:PUT</b> .....	<b>184</b>
<b>115 PUT /email/{domain}/usernames/{mailboxName}/webmail</b> .....	<b>185</b>
115.1 Contents.....	185
115.2 Request.....	185
115.3 Response.....	185
115.4 Examples.....	186

# Table of Contents

<b>116 Email-secretQuestions:GET</b> .....	<b>188</b>
<b>117 GET /email/secretQuestions</b> .....	<b>189</b>
117.1 Contents.....	189
117.2 Request.....	189
117.3 Response.....	189
117.4 Examples.....	189
<b>118 Email-secretQuestionSettings:GET</b> .....	<b>191</b>
<b>119 GET /email/secretQuestionSettings</b> .....	<b>192</b>
119.1 Contents.....	192
119.2 Request.....	192
119.3 Response.....	192
119.4 Examples.....	192
<b>120 Template:ResetPassword</b> .....	<b>193</b>
<b>121 POST {{{base}}}/email/{domain}/usernames/{mailboxName}/{{{resource}}}/action</b> .....	<b>194</b>
121.1 Contents.....	194
121.2 Request.....	194
121.3 Response.....	195
121.4 Examples.....	195
121.5 See also.....	195

# 1 Accounts-account-email-domain-aliases:GET



## 2 GET /accounts/{account}/email/{domain}/aliases

Retrieves a list of aliases for a specific account and domain

### 2.1 Contents

- 1 GET /accounts/{account}/email/{domain}/aliases
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
  - ◆ 1.4 See also

### 2.2 Request

GET /accounts/{account}/email/{domain}/aliases

#### 2.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes  
domain - *string*  
The specific domain for which the call will retrieve the list

### 2.3 Response

#### 2.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or account does not exist.

#### 2.3.2 Response Body

```
{
  "list": [
    "{alias}"
  ],
  "links": [
    {
      "href": "{href}",
      "rel": "{rel}"
    }
  ]
}
```

##### 2.3.2.1 Parameters

alias - *string*  
Domain name alias

href - *string*  
Link to other resources relevant to the mailboxes lists

rel - *string*  
Type of relation to the resource for the provided link

### 2.4 Examples

#### 2.4.1 Success scenario

##### Request

GET accounts/test-account123/email/test.com/aliases

##### Response

```
{
```

```
"list": [
  "test-alias.com", "test-alias-2.com"
],
"links": [
  {
    "href": "https://api.hostway.com/accounts/test-account123/email/test.com/aliases/",
    "rel": "self"
  }
]
}
```

## 2.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

### **3 Accounts-account-email-domain-blackwhitelist-address:DELETE**

## 4 DELETE /accounts/{account}/email/{domain}/blackwhitelist/{address}

Remove black/white listed address

### 4.1 Contents

- 1 DELETE  
/accounts/{account}/email/{domain}/blackwhitelist/{address>
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Delete address from black/white list

### 4.2 Request

```
DELETE /accounts/{account}/email/{domain}/blackwhitelist/{address}
```

#### 4.2.1 URI Parameters

account - *string*  
domain - *string*  
address - *string*

#### 4.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 4.3 Response

#### 4.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or address does not exist.

### 4.4 Examples

#### 4.4.1 Delete address from black/white list

##### Request

```
DELETE /accounts/account-number/email/test.com/blackwhitelist/mb1-white20150114-0113@somedomain.com
```

##### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 5 Accounts-account-email-domain-blackwhitelist-address:GET

## 6 GET /accounts/{account}/email/{domain}/blackwhitelist/{address}

Get black/white listed address details

### 6.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/blackwhitelist/{address>
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Get address details

### 6.2 Request

GET /accounts/{account}/email/{domain}/blackwhitelist/{address}

#### 6.2.1 URI Parameters

account - *string*  
domain - *string*  
address - *string*

#### 6.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 6.3 Response

#### 6.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 6.3.2 Response Body

```
{
  "type": "{type}",
  "address": "{address}",
  "links": {links}
}
```

##### 6.3.2.1 Parameters

type - *string*  
Type of address included in black/white list. Possible values are "W" or "B". "W" means the mailbox is included in "whitelist", "B" - included in "blacklist".

address - *string*  
The address of the black/whitelisted mailbox or domain.

links - *list*  
*Hypermedia* to the get black/whitelisted address

### 6.4 Examples

#### 6.4.1 Get address details

##### Request

GET /accounts/account-number/email/test.com/blackwhitelist/mb1-white20150114-01113@somedomain.com

##### Response

```
200 OK
{
  "type": "W",
```

```
"address": "mb1-white20150114-0113@somedomain.com",
"links": [{
  "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist/mb1-white20150114-0113@somedomain.com/",
  "rel": "self"
}]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 7 Accounts-account-email-domain-blackwhitelist:GET



## 8 GET /accounts/{account}/email/{domain}/blackwhitelist

Retrieves a list of all addresses included in black/white list for a specific domain

### 8.1 Contents

- 1 GET /accounts/{account}/email/{domain}/blackwhitelist
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - ◇ 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Get all addresses included in black/white list
    - ◇ 1.3.2 Get filtered list by address
    - ◇ 1.3.3 Get filtered list by type
    - ◇ 1.3.4 Paginated request
    - ◇ 1.3.5 Sorted request

### 8.2 Request

GET /accounts/{account}/email/{domain}/blackwhitelist

#### 8.2.1 URI Parameters

account - *string*  
domain - *string*

#### 8.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 8.3 Response

#### 8.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 8.3.2 Response Body

```
{
  "totalCount": {totalCount},
  "list": [
    {
      "type": "{type}",
      "address": "{address}"
    }
  ],
  "links": [
    {
      "href": "{href}",
      "rel": "{rel}"
    }
  ]
}
```

##### 8.3.2.1 Parameters

totalCount - *int*  
The total count of addresses included in black/white list

type - *string*  
Type of address included in black/white list. Possible values are "W" or "B". "W" means the mailbox is included in "whitelist", "B" - included in "blacklist".

address - *string*  
The address of the black/whitelisted mailbox or domain.

links - *list* [Hypermedia](#) to the get black/whitelisted addresses

## 8.4 Examples

### 8.4.1 Get all addresses included in black/white list

#### Request

```
GET /accounts/account-number/email/test.com/blackwhitelist
```

#### Response

200 OK

```
{
  "totalCount": 4,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-black@domain.com"
    },
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    },
    {
      "type": "B",
      "address": "@sometestdomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist",
      "rel": "self"
    }
  ]
}
```

### 8.4.2 Get filtered list by address

#### Request

```
GET /accounts/account-number/email/test.com/blackwhitelist?filterAddress=*somedomain.com*
```

#### Response

200 OK

```
{
  "totalCount": 2,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist?filterAddress=*somedomain.com*",
      "rel": "self"
    }
  ]
}
```

### 8.4.3 Get filtered list by type

#### Request

```
GET /accounts/account-number/email/test.com/blackwhitelist?filterType=W
```

#### Response

200 OK

```
{
  "totalCount": 1,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist?filterType=W",
      "rel": "self"
    }
  ]
}
```

```
} ]
```

## 8.4.4 Paginated request

### Request

```
GET /accounts/account-number/email/test.com/blackwhitelist?pageSize=2&page=1
```

### Response

```
200 OK
```

```
{
  "totalCount": 2,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-white@domain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist?pageSize=2&page=2",
      "rel": "next"
    },
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist?pageSize=2&page=2",
      "rel": "last"
    }
  ]
}
```

## 8.4.5 Sorted request

### Request

```
GET /accounts/account-number/email/test.com/blackwhitelist?sortField=address&sortOrder=desc
```

### Response

```
200 OK
```

```
{
  "totalCount": 4,
  "list": [
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-black@domain.com"
    },
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "@sometestdomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/blackwhitelist?sortField=address&sortOrder=desc",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 9 Accounts-account-email-domain-blackwhitelist:PUT

## 10 PUT /accounts/{account}/email/{domain}/blackwhitelist

Adding/updating addresses to black/white list for a specific domain

### 10.1 Contents

- 1 PUT  
/accounts/{account}/email/{domain}/blackwhitelist
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure adding address with missing required parameter
    - ◇ 1.3.2 Failure adding address with not valid address and type
    - ◇ 1.3.3 Success adding email addresses to blackwhitelist
    - ◇ 1.3.4 Success adding domain addresses to blackwhitelist

### 10.2 Request

PUT /accounts/{account}/email/{domain}/blackwhitelist

#### 10.2.1 URI Parameters

account - *string*

domain - *string*

#### 10.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The *Authentication* credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

#### 10.2.3 Request Body

```
{
  "list": [
    {
      "address": "{address}",
      "type": "{type}"
    }
  ]
}
```

##### 10.2.3.1 Parameters

address - *string*

Must be a valid email or domain address. Required.

type - *string*

One of W, B. Required.

### 10.3 Response

#### 10.3.1 Status Code

204 No Content

Success

400 Bad Request

The format of the request body is invalid or the address does not meet the requirements.

401 Unauthorized

The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found

The domain name does not exist.

### 10.4 Examples

#### 10.4.1 Failure adding address with missing required parameter

Request

```
PUT /accounts/account-number/email/test.com/blackwhitelist
{"list":[{"address":"mb1-white@somedomain.com"}]}
```

## Response

400 Bad Request

```
{
  "badRequest": {
    "message": "POST data error",
    "code": 400,
    "details": {
      "list.0.type": "Required"
    }
  }
}
```

## 10.4.2 Failure adding address with not valid address and type

### Request

```
PUT /accounts/account-number/email/test.com/blackwhitelist
{"list":[{"address":"somedomain.com","type":"A"}]}
```

### Response

400 Bad Request

```
{
  "badRequest": {
    "message": "POST data error",
    "code": 400,
    "details": {
      "list.0.type": "\"A\" is not one of B, W",
      "list.0.address": "Invalid address format: somedomain.com"
    }
  }
}
```

## 10.4.3 Success adding email addresses to blackwhitelist

### Request

```
PUT /accounts/account-number/email/test.com/blackwhitelist
{"list":[{"address":"mb1-white20150114-0113@somedomain.com", "type":"W"}, {"address":"mb2-white20150114-0113@somedomain.com", "type":"W"}]}
```

### Response

204 No Content

## 10.4.4 Success adding domain addresses to blackwhitelist

### Request

```
PUT /accounts/account-number/email/test.com/blackwhitelist
{
  "list": [
    {
      "type": "W",
      "address": "@domainaddr1.com"
    },
    {
      "type": "B",
      "address": "@someotherdomain.com"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 11 Accounts-account-email-domain-cleanmailplus:GET

## 12 GET accounts/{account}/email/{domain}/cleanmailplus

Gets domain's CleanMailPlus info.

### 12.1 Contents

- 1 GET accounts/{account}/email/{domain}/cleanmailplus
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure getting CleanMailPlus info for a non-existent domain
    - ◇ 1.3.2 Success getting CleanMailPlus info

### 12.2 Request

GET accounts/{account}/email/{domain}/cleanmailplus

#### 12.2.1 URI Parameters

account - *string*  
domain - *string*

#### 12.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 12.3 Response

#### 12.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 12.3.2 Response Body

```
[
  {
    "selected": "{selected_option_value}",
    "fieldName": "{field_name}",
    "description": "{description}",
    "options": [
      {
        "name": "{option_name}",
        "value": "{option_value}"
      }
    ]
  }
]
```

##### 12.3.2.1 Parameters

selected\_option\_value - *string*  
Currently selected option value.

field\_name - *string*  
Field name.

description - *string*  
Descriptive option info.

options - *list*  
List of options.

options.name - *string*  
Option's name.

options.value - *string*  
Option's value.



## 12.4 Examples

### 12.4.1 Failure getting CleanMailPlus info for a non-existent domain

#### Request

```
GET accounts/test-account123/email/non-existent-domain.com/cleanmailplus
```

#### Response

```
404 Not Found
```

### 12.4.2 Success getting CleanMailPlus info

#### Request

```
GET accounts/test-account123/email/test.com/cleanmailplus
```

#### Response

```
200 OK
```

```
[
  {
    "selected": "6",
    "fieldName": "SPAM_LEVEL",
    "description": "SPAM Tag Level",
    "options": [
      {
        "name": "Least Aggressive",
        "value": "6"
      },
      {
        "name": "Less Aggressive",
        "value": "5"
      },
      {
        "name": "Normal",
        "value": "4"
      },
      {
        "name": "More Aggressive",
        "value": "3"
      },
      {
        "name": "Most Aggressive",
        "value": "2"
      },
      {
        "name": "Ultra Aggressive",
        "value": "1"
      }
    ]
  },
  {
    "selected": "2",
    "fieldName": "SPAM_ACTION",
    "description": "SPAM Detected Action",
    "options": [
      {
        "name": "None",
        "value": "0"
      },
      {
        "name": "Tag Subject",
        "value": "2"
      },
      {
        "name": "Move to SPAM Folder",
        "value": "1"
      },
      {
        "name": "Delete",
        "value": "3"
      }
    ]
  },
  {
    "selected": "1",
    "fieldName": "ATTACHMENT_CONTROL",
    "description": "Attachment Control",
    "options": [
      {
        "name": "Enable",
        "value": "1"
      },
      {
        "name": "Disable",
        "value": "0"
      }
    ]
  },
  {
    "selected": "1",
    "fieldName": "VIRUS_ACTION",
    "description": "Virus Detected Action",
    "options": [
      {
        "name": "No Virus Filter",
        "value": "0"
      }
    ]
  }
]
```

```

    },
    {
      "name": "Delete Automatically",
      "value": "1"
    }
  ]
},
{
  "selected": "0",
  "fieldName": "SPAM_LEARNING",
  "description": "SPAM Learning Status",
  "options": []
},
{
  "selected": "20",
  "fieldName": "SPAM_DISCARD_LEVEL",
  "description": "SPAM Discard Level",
  "options": []
},
{
  "selected": "0",
  "fieldName": "PHISHING_CHECKS",
  "description": "Phishing Checks",
  "options": [
    {
      "name": "Enabled",
      "value": "1"
    },
    {
      "name": "Disabled",
      "value": "0"
    }
  ]
},
{
  "selected": "0",
  "fieldName": "CLEANMAIL_PLUS",
  "description": "CleanMail Plus Enable",
  "options": [
    {
      "name": "Enable",
      "value": "1"
    },
    {
      "name": "Disable",
      "value": "0"
    }
  ]
},
{
  "selected": "1",
  "fieldName": "SPAM_CHECK",
  "description": "SPAM Check Option",
  "options": [
    {
      "name": "Enable",
      "value": "1"
    },
    {
      "name": "Disable",
      "value": "0"
    }
  ]
}
]
}
]

```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 13 Accounts-account-email-domain-cleanmailplus:PUT

## 14 PUT accounts/{account}/email/{domain}/cleanmailplus

Updates domain's CleanMailPlus info.

### 14.1 Contents

- 1 PUT accounts/{account}/email/{domain}/cleanmailplus
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating the CleanMailPlus filter attributes for a non-existent domain
    - ◇ 1.3.2 Failure updating the CleanMailPlus filter attributes by providing bad request body
    - ◇ 1.3.3 Failure updating the cleanmailplus filter attribute by providing a non-existent attribute name
    - ◇ 1.3.4 Failure updating the cleanmailplus filter attribute by providing a non-valid attribute value
    - ◇ 1.3.5 Success updating the CleanMailPlus filter attributes

### 14.2 Request

PUT accounts/{account}/email/{domain}/cleanmailplus

#### 14.2.1 URI Parameters

account - *string*

domain - *string*

#### 14.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The *Authentication* credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

#### 14.2.3 Request Body

```
{
  "attributes": [
    {
      "attributeName": "{attribute_name}",
      "attributeValue": "{attribute_value}"
    }
  ]
}
```

##### 14.2.3.1 Parameters

attributeName - *string*

Attribute's name.

attributeValue - *string*

New attribute's value.

### 14.3 Response

#### 14.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

### 14.4 Examples

#### 14.4.1 Failure updating the CleanMailPlus filter attributes for a non-existent domain

##### Request

PUT accounts/{account}/email/non-existent-domain.com/cleanmailplus

```
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "1"
    }
  ]
}
```

### Response

```
404 Not Found
{
  "itemNotFound": {
    "message": "Resource not found",
    "code": 404,
    "details": {
      "message": "Domain not found"
    }
  }
}
```

## 14.4.2 Failure updating the CleanMailPlus filter attributes by providing bad request body

### Request

```
PUT accounts/{account}/email/test.com/cleanmailplus
{
  "badBody": "badBodyValue"
}
```

### Response

```
400 Bad Request
{
  "attributes": "Required"
}
```

## 14.4.3 Failure updating the cleanmailplus filter attribute by providing a non-existent attribute name

### Request

```
PUT accounts/{account}/email/test.com/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "NON_EXISTENT_ATTRIBUTE_NAME",
      "attributeValue": "1"
    }
  ]
}
```

### Response

```
400 Bad Request
{
  "attributes.0": "NON_EXISTENT_ATTRIBUTE_NAME is not valid filter attribute name"
}
```

## 14.4.4 Failure updating the cleanmailplus filter attribute by providing a non-valid attribute value

### Request

```
PUT accounts/{account}/email/test.com/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "val01"
    }
  ]
}
```

### Response

```
400 Bad Request
{
  "attributes.0": "val01 is not valid attribute option for SPAM_LEVEL filter attribute"
}
```

## 14.4.5 Success updating the CleanMailPlus filter attributes

### Request

```
PUT accounts/{account}/email/test.com/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "1"
    },
    {

```

```
    "attributeName": "SPAM_ACTION",  
    "attributeValue": "1"  
  }  
]  
}
```

## Response

204 No Content

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 15 Accounts-account-email-domain-contacts-contact:DELETE

## 16 DELETE /accounts/{account}/email/{domain}/contacts/{contact}

Deletes domain's global address book contact.

### 16.1 Contents

- 1 DELETE  
/accounts/{account}/email/{domain}/contacts/{contact}
- ◆ 1.1 Request
  - ◇ 1.1.1 URI Parameters
  - ◇ 1.1.2 Request Headers
- ◆ 1.2 Response
  - ◇ 1.2.1 Status Code
- ◆ 1.3 Examples
  - ◇ 1.3.1 Success deleting domain's global address book contact
- ◆ 1.4 See also

### 16.2 Request

DELETE /accounts/{account}/email/{domain}/contacts/{contact}

#### 16.2.1 URI Parameters

account - *string*  
domain - *string*  
contact - *string*

#### 16.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 16.3 Response

#### 16.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain or contact does not exist.

### 16.4 Examples

#### 16.4.1 Success deleting domain's global address book contact

##### Request

```
DELETE https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact
```

##### Response

```
204 No Content
```

### 16.5 See also

- [Global Address Book Public API](#)



## 17 Accounts-account-email-domain-contacts-contact:GET

## 18 GET /accounts/{account}/email/{domain}/contacts/{contact}

Gets domain's global address book contact information

### 18.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/contacts/{contact}
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Status Code
    - ◇ 1.1.4 Response Body
      - 1.1.4.1 Parameters
  - ◆ 1.2 Examples
    - ◇ 1.2.1 Error getting global address book contact info
    - ◇ 1.2.2 Success getting global address book contact info
  - ◆ 1.3 See also

### 18.2 Request

GET /accounts/{account}/email/{domain}/contacts/{contact}

#### 18.2.1 URI Parameters

account - *string*  
domain - *string*  
contact - *string*

#### 18.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 18.2.3 Status Code

200 Ok  
Success

400 Bad Request  
The format of the request body is invalid or the parameters don't meet the requirements.

404 Not Found  
The domain name or contact does not exist.

#### 18.2.4 Response Body

```
{
  "name": "{name}",
  "email": "{email}",
  "firstName": "{firstName}",
  "lastName": "{lastName}",
  "displayName": "{displayName}"
  "title": "{title}",
  "homePhone": "{homePhone}",
  "mobilePhone": "{mobilePhone}",
  "businessPhone": "{businessPhone}",
  "businessFax": "{businessFax}",
  "isdnNumber": "{isdnNumber}",
  "pager": "{pager}",
  "telexNumber": "{telexNumber}",
  "company": "{company}",
  "department": "{department}",
  "roomNumber": "{roomNumber}",
  "streetAddress": "{streetAddress}",
  "postalCode": "{postalCode}",
  "city": "{city}",
  "state": "{state}",
  "country": "{country}",
  "links": [{"href": "{APIBaseURL}/accounts/{account}/email/{domain}/contacts/{name}", "rel": "self"}]
}
```

##### 18.2.4.1 Parameters

name - *string*  
The name of global address book contact.

email - *string*  
The email of global address book contact.

firstName - *string*  
The first name of global address book contact.

lastName - *string*  
The last name of global address book contact.

displayName - *string*  
The display name of global address book contact.

title - *string*  
The title of global address book contact person.

homePhone - *string*  
The home phone of global address book contact.

mobilePhone - *string*  
The home phone of global address book contact.

businessPhone - *string*  
The business phone of global address book contact.

businessFax - *string*  
The business fax of global address book contact.

isdnNumber - *string*  
The isdn number of global address book contact.

pager - *string*  
The pager of global address book contact.

telexNumber - *string*  
The telex number of global address book contact.

company - *string*  
The company name of global address book contact.

department - *string*  
The department name of global address book contact.

roomNumber - *string*  
The room number of global address book contact.

streetAddress - *string*  
The street address of global address book contact.

postalCode - *string*  
The postal code of global address book contact.

city - *string*  
The city name of global address book contact.

state - *string*  
The state name of global address book contact.

country - *string*  
The country name of global address book contact.

## 18.3 Examples

### 18.3.1 Error getting global address book contact info

#### Request

```
GET https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/non-existent-contact
```

#### Response

```
404 Not Found
```

### 18.3.2 Success getting global address book contact info

#### Request

```
GET https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact
```

#### Response

```
200 Ok
{
  "city": "",
  "businessPhone": "",
  "displayName": "Display Name",
  "mobilePhone": "",
  "firstName": "First",
  "title": "",
  "roomNumber": "",
  "lastName": "Last",
  "company": ""
}
```

```
"pager": "",
"state": "",
"department": "",
"streetAddress": "",
"isdnNumber": "",
"postalCode": "",
"country": "",
"telexNumber": "",
"businessFax": "",
"email": "test@test.com",
"homePhone": "",
"links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact", "rel": "self"}]
}
```

## 18.4 See also

- [Global Address Book Public API](#)

## 19 Accounts-account-email-domain-contacts-contact:PUT

## 20 PUT /accounts/{account}/email/{domain}/contacts/{contact}

Updates domain's global address book contact info

### 20.1 Contents

- 1 PUT /accounts/{account}/email/{domain}/contacts/{contact}
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating global address book contact because of erroneous data
    - ◇ 1.3.2 Success updating global address book contact
  - ◆ 1.4 See also

### 20.2 Request

PUT /accounts/{account}/email/{domain}/contacts/{contact}

#### 20.2.1 URI Parameters

account - *string*  
domain - *string*  
contact - *string*

#### 20.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 20.2.3 Request Body

```
{
  "email": "{email}",
  "firstName": "{firstName}",
  "lastName": "{lastName}",
  "displayName": "{displayName}"
  "title": "{title}",
  "homePhone": "{homePhone}",
  "mobilePhone": "{mobilePhone}",
  "businessPhone": "{businessPhone}",
  "businessFax": "{businessFax}",
  "isdnNumber": "{isdnNumber}",
  "pager": "{pager}",
  "telexNumber": "{telexNumber}",
  "company": "{company}",
  "department": "{department}",
  "roomNumber": "{roomNumber}",
  "streetAddress": "{streetAddress}",
  "postalCode": "{postalCode}",
  "city": "{city}",
  "state": "{state}",
  "country": "{country}"
}
```

##### 20.2.3.1 Parameters

email - *string*  
The email of global address book contact.

firstName - *string*  
The first name of global address book contact.

lastName - *string*  
The last name of global address book contact.

displayName - *string*  
The display name of global address book contact.

title - *string*  
The title of global address book contact person.

homePhone - *string*  
The home phone of global address book contact.

mobilePhone - *string*  
The home phone of global address book contact.

businessPhone - *string*  
The business phone of global address book contact.

businessFax - *string*  
The business fax of global address book contact.

isdnNumber - *string*  
The isdn number of global address book contact.

pager - *string*  
The pager of global address book contact.

telexNumber - *string*  
The telex number of global address book contact.

company - *string*  
The company name of global address book contact.

department - *string*  
The department name of global address book contact.

roomNumber - *string*  
The room number of global address book contact.

streetAddress - *string*  
The street address of global address book contact.

postalCode - *string*  
The postal code of global address book contact.

city - *string*  
The city name of global address book contact.

state - *string*  
The state name of global address book contact.

country - *string*  
The country name of global address book contact.

## 20.3 Response

### 20.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the parameters don't meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

## 20.4 Examples

### 20.4.1 Failure updating global address book contact because of erroneous data

#### Request

```
PUT https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact
{
  "email": "updated-email",
  "displayName": "Display Name Updated"
}
```

#### Response

```
400 Bad Request
{
  "email": "Invalid email address"
}
```

### 20.4.2 Success updating global address book contact

#### Request

```
PUT https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact
{
  "email": "new@test.com",
  "mobilePhone": "123456789"
}
```

#### Response

204 No Content

## 20.5 See also

- [Global Address Book Public API](#)



## 21 Accounts-account-email-domain-contacts:GET

## 22 GET

**/accounts/{account}/email/{domain}/contacts?filter{FieldName}={FieldValue}[&filter{Fi**

Filter domain's global address book contacts list

## 23 GET

### /accounts/{account}/email/{domain}/contacts?search{FieldName}={FieldValue}[&search

Search domain's global address book contacts list

#### 23.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/contacts?filter{FieldName}={FieldValue}[&filter{FieldName}={FieldValue}]
- 2 GET  
/accounts/{account}/email/{domain}/contacts?search{FieldName}={FieldValue}[&search{FieldName}={FieldValue}]
  - ◆ 2.1 Request
    - ◇ 2.1.1 URI Parameters
    - ◇ 2.1.2 Request Headers
    - ◇ 2.1.3 Status Code
    - ◇ 2.1.4 Response Body
      - 2.1.4.1 Parameters
  - ◆ 2.2 Examples
    - ◇ 2.2.1 Success filtering global address book contacts with filterName and filterEmail parameters
    - ◇ 2.2.2 Success filtering global address book contacts with only filterEmail parameter
    - ◇ 2.2.3 Success searching global address book contacts with only searchName parameter
    - ◇ 2.2.4 Success filtering global address book contacts with non-existent filterName parameter
    - ◇ 2.2.5 Success searching global address book contacts with non-existent searchName parameter
  - ◆ 2.3 See also

#### 23.2 Request

GET /accounts/{account}/email/{domain}/contacts?filter{FieldName}={FieldValue} [&filter{FieldName}={FieldValue}]

GET /accounts/{account}/email/{domain}/contacts?search{FieldName}={FieldValue} [&search{FieldName}={FieldValue}]

##### 23.2.1 URI Parameters

account - *string*

domain - *string*

filter{FieldName} - *string*

Specifies fields to filter by, i.e. filterName, filterEmail, etc...

search{FieldName} - *string*

Specifies fields to search by, i.e. searchName, searchEmail, etc...

##### 23.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The [Authentication](#) credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

##### 23.2.3 Status Code

200 Ok

Success

404 Not Found

The domain name does not exist.

##### 23.2.4 Response Body

```
{
  "count": {count},
  "list": [
    {
      "name": "{name}",
      "email": "{email}",
      "firstName": "{firstName}",
      "lastName": "{lastName}",
      "displayName": "{displayName}"
      "title": "{title}",
      "homePhone": "{homePhone}",
      "mobilePhone": "{mobilePhone}",
      "businessPhone": "{businessPhone}",
      "businessFax": "{businessFax}",
      "isdnNumber": "{isdnNumber}",
      "pager": "{pager}",
      "telexNumber": "{telexNumber}",
      "company": "{company}",
      "department": "{department}",
      "roomNumber": "{roomNumber}",
      "streetAddress": "{streetAddress}",
      "postalCode": "{postalCode}",
      "city": "{city}",
      "state": "{state}",
      "country": "{country}",
      "links": [{"href": "{APIBaseUrl}/accounts/{account}/email/{domain}/contacts/{name}", "rel": "self"}],
    },
  ],
  "links": [{"href": "{APIBaseUrl}/accounts/{account}/email/{domain}/contacts", "rel": "self"}],
}
```

```
}
```

### 23.2.4.1 Parameters

count - *int*  
The count of global address book contacts matching the serch/filter condition.

name - *string*  
The name of global address book contact.

email - *string*  
The email of global address book contact.

firstName - *string*  
The first name of global address book contact.

lastName - *string*  
The last name of global address book contact.

displayName - *string*  
The display name of global address book contact.

title - *string*  
The title of global address book contact person.

homePhone - *string*  
The home phone of global address book contact.

mobilePhone - *string*  
The home phone of global address book contact.

businessPhone - *string*  
The business phone of global address book contact.

businessFax - *string*  
The business fax of global address book contact.

isdnNumber - *string*  
The isdn number of global address book contact.

pager - *string*  
The pager of global address book contact.

telexNumber - *string*  
The telex number of global address book contact.

company - *string*  
The company name of global address book contact.

department - *string*  
The department name of global address book contact.

roomNumber - *string*  
The room number of global address book contact.

streetAddress - *string*  
The street address of global address book contact.

postalCode - *string*  
The postal code of global address book contact.

city - *string*  
The city name of global address book contact.

state - *string*  
The state name of global address book contact.

country - *string*  
The country name of global address book contact.

## 23.3 Examples

### 23.3.1 Success filtering global address book contacts with filterName and filterEmail parameters

#### Request

```
GET https://api.hostway.com/accounts/account007/email/test-domain.com/contacts?filterName=search-contact&filterEmail=new@test.com
```

#### Response

```
200 Ok
{
  "count": 1,
  "list": [
    {
      "city": "",
      "businessPhone": "",

```

```

        "displayName": "Display Name",
        "mobilePhone": "",
        "firstName": "First",
        "title": "",
        "roomNumber": "",
        "lastName": "Last",
        "company": "",
        "pager": "",
        "state": "",
        "department": "",
        "streetAddress": "",
        "isdnNumber": "",
        "postalCode": "",
        "country": "",
        "telexNumber": "",
        "businessFax": "",
        "email": "new@test.com",
        "homePhone": "",
        "name": "search-contact",
        "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/search-contact", "rel": "self"}],
    },
    "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts", "rel": "self"}],
}

```

### 23.3.2 Success filtering global address book contacts with only filterEmail parameter

#### Request

GET <https://api.hostway.com/accounts/account007/email/test-domain.com/contacts?filterEmail=new@test.com>

#### Response

```

200 Ok
{
  "count": 2,
  "list": [
    {
      "city": "",
      "businessPhone": "",
      "displayName": "Display Name",
      "mobilePhone": "",
      "firstName": "First",
      "title": "",
      "roomNumber": "",
      "lastName": "Last",
      "company": "",
      "pager": "",
      "state": "",
      "department": "",
      "streetAddress": "",
      "isdnNumber": "",
      "postalCode": "",
      "country": "",
      "telexNumber": "",
      "businessFax": "",
      "email": "new@test.com",
      "homePhone": "",
      "name": "search-contact",
      "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/search-contact", "rel": "self"}],
    },
    {
      "city": "",
      "businessPhone": "",
      "displayName": "Display Name",
      "mobilePhone": "123456789",
      "firstName": "First",
      "title": "",
      "roomNumber": "",
      "lastName": "Last",
      "company": "",
      "pager": "",
      "state": "",
      "department": "",
      "streetAddress": "",
      "isdnNumber": "",
      "postalCode": "",
      "country": "",
      "telexNumber": "",
      "businessFax": "",
      "email": "new@test.com",
      "homePhone": "",
      "name": "test-contact",
      "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact", "rel": "self"}],
    }
  ],
  "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts", "rel": "self"}],
}

```

### 23.3.3 Success searching global address book contacts with only searchName parameter

#### Request

GET <https://api.hostway.com/accounts/account007/email/test-domain.com/contacts?searchName=contact>

#### Response

```

200 Ok
{
  "count": 2,
  "list": [
    {
      "city": "",
      "businessPhone": "",
      "displayName": "Display Name",
      "mobilePhone": "",
      "firstName": "First",
      "title": "",
      "roomNumber": "",
      "lastName": "Last",
      "company": "",
      "pager": "",
      "state": "",
      "department": "",
      "streetAddress": "",
      "isdnNumber": "",
      "postalCode": "",
      "country": "",
      "telexNumber": "",
      "businessFax": "",
      "email": "new@test.com",
      "homePhone": "",
      "name": "search-contact",
      "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/search-contact", "rel": "self"}]
    },
    {
      "city": "",
      "businessPhone": "",
      "displayName": "Display Name",
      "mobilePhone": "123456789",
      "firstName": "First",
      "title": "",
      "roomNumber": "",
      "lastName": "Last",
      "company": "",
      "pager": "",
      "state": "",
      "department": "",
      "streetAddress": "",
      "isdnNumber": "",
      "postalCode": "",
      "country": "",
      "telexNumber": "",
      "businessFax": "",
      "email": "new@test.com",
      "homePhone": "",
      "name": "test-contact",
      "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts/test-contact", "rel": "self"}]
    }
  ],
  "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts", "rel": "self"}]
}

```

### 23.3.4 Success filtering global address book contacts with non-existent filterName parameter

#### Request

```
GET https://api.hostway.com/accounts/account007/email/test-domain.com/contacts?filterName=notexisting
```

#### Response

```

200 Ok
{
  "count": 0,
  "list": [],
  "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts", "rel": "self"}]
}

```

### 23.3.5 Success searching global address book contacts with non-existent searchName parameter

#### Request

```
GET https://api.hostway.com/accounts/account007/email/test-domain.com/contacts?searchName=notexisting
```

#### Response

```

200 Ok
{
  "count": 0,
  "list": [],
  "links": [{"href": "https://api.hostway.com/accounts/account007/email/test-domain.com/contacts", "rel": "self"}]
}

```

## 23.4 See also

- [Global Address Book Public API](#)

## 24 Accounts-account-email-domain-contacts:POST

## 25 POST /accounts/{account}/email/{domain}/contacts

Creates global address book contact for domain

### 25.1 Contents

- 1 POST /accounts/{account}/email/{domain}/contacts
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure creating global address book contact because of missing data
    - ◇ 1.3.2 Failure creating global address book contact because of erroneous data
    - ◇ 1.3.3 Success creating global address book contact
    - ◇ 1.3.4 Failure creating contact because contact with the same name already exists
  - ◆ 1.4 See also

### 25.2 Request

POST /accounts/{account}/email/{domain}/contacts

#### 25.2.1 URI Parameters

account - *string*

domain - *string*

#### 25.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The *Authentication* credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

#### 25.2.3 Request Body

```
{
  "name": "{name}",
  "email": "{email}",
  "firstName": "{firstName}",
  "lastName": "{lastName}",
  "displayName": "{displayName}"
  "title": "{title}",
  "homePhone": "{homePhone}",
  "mobilePhone": "{mobilePhone}",
  "businessPhone": "{businessPhone}",
  "businessFax": "{businessFax}",
  "isdnNumber": "{isdnNumber}",
  "pager": "{pager}",
  "telexNumber": "{telexNumber}",
  "company": "{company}",
  "department": "{department}",
  "roomNumber": "{roomNumber}",
  "streetAddress": "{streetAddress}",
  "postalCode": "{postalCode}",
  "city": "{city}",
  "state": "{state}",
  "country": "{country}"
}
```

##### 25.2.3.1 Parameters

name - *string*

The name of global address book contact. Required.

email - *string*

The email of global address book contact. Required.

firstName - *string*

The first name of global address book contact.

lastName - *string*

The last name of global address book contact.

displayName - *string*

The display name of global address book contact. Required.

title - *string*

The title of global address book contact person.



homePhone - *string*  
The home phone of global address book contact.

mobilePhone - *string*  
The home phone of global address book contact.

businessPhone - *string*  
The business phone of global address book contact.

businessFax - *string*  
The business fax of global address book contact.

isdnNumber - *string*  
The isdn number of global address book contact.

pager - *string*  
The pager of global address book contact.

telexNumber - *string*  
The telex number of global address book contact.

company - *string*  
The company name of global address book contact.

department - *string*  
The department name of global address book contact.

roomNumber - *string*  
The room number of global address book contact.

streetAddress - *string*  
The street address of global address book contact.

postalCode - *string*  
The postal code of global address book contact.

city - *string*  
The city name of global address book contact.

state - *string*  
The state name of global address book contact.

country - *string*  
The country name of global address book contact.

## 25.3 Response

### 25.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the parameters don't meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

409 Conflict  
The global address book contact already exists.

## 25.4 Examples

### 25.4.1 Failure creating global address book contact because of missing data

#### Request

```
POST https://api.hostway.com/accounts/account007/email/test-domain.com/contacts
{ }
```

#### Response

```
400 Bad Request
{
  "displayName": "Required",
  "name": "Required",
  "email": "Required"
}
```

### 25.4.2 Failure creating global address book contact because of erroneous data

#### Request

```
POST https://api.hostway.com/accounts/account007/email/test-domain.com/contacts
{
  "name": "test-contact",
}
```

```
    "email": "test",
    "displayName": "Display Name"
}
```

## Response

400 Bad Request

```
{
  "email": "Invalid email address"
}
```

### 25.4.3 Success creating global address book contact

#### Request

POST <https://api.hostway.com/accounts/account007/email/test-domain.com/contacts>

```
{
  "name": "test-contact",
  "email": "test@test.com",
  "firstName": "First",
  "lastName": "Last",
  "displayName": "Display Name"
}
```

#### Response

204 No Content

### 25.4.4 Failure creating contact because contact with the same name already exists

#### Request

POST <https://api.hostway.com/accounts/account007/email/test-domain.com/contacts>

```
{
  "name": "test-contact",
  "email": "test@test.com",
  "firstName": "First",
  "lastName": "Last",
  "displayName": "Display Name"
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "guid": "af4f4c85-4c8f-4c63-95d1-f3fb8cb8c479",
    "code": 409,
    "message": "Data conflict"
  }
}
```

## 25.5 See also

- [Global Address Book Public API](#)

## 26 Accounts-account-email-domain-infostore:GET

## 27 GET /accounts/{account}/email/{domain}/infostore

Gets InfoStore usage.

### 27.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/infostore
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting account type

### 27.2 Request

GET /accounts/{account}/email/{domain}/infostore

#### 27.2.1 URI Parameters

account - *string*  
domain - *string*

#### 27.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 27.3 Response

#### 27.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 27.3.2 Response Body

```
{
  "infoStore": {
    "usage": 0,
    "quota": 20
  },
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/infostore/",
      "rel": "self"
    }
  ]
}
```

##### 27.3.2.1 Parameters

usage - *int*  
InfoStore usage in MB.

quota - *int*  
InfoStore quota in MB.

### 27.4 Examples

#### 27.4.1 Success getting account type

##### Request

GET /accounts/test/email/test.com/infostore

## Response

```
200 OK  
{ "infoStore": { "usage": 0, "quota": 20 }, "links": [ { "href": "{APIBaseURL}/accounts/test/email/test.com/infostore/", "rel": "self" } ] }
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

# 28 Accounts-account-email-domain-infostore:PUT

= PUT /accounts/{account}/email/{domain}/infostore Change InfoStore quota.

## 28.1 Contents

- 1 Request
  - ◆ 1.1 URI Parameters
  - ◆ 1.2 Request Headers
  - ◆ 1.3 Request Body
    - ◇ 1.3.1 Parameters
- 2 Response
  - ◆ 2.1 Status Code
- 3 Examples
  - ◆ 3.1 Failure setting new InfoStore quota with missing required parameter
  - ◆ 3.2 Failure changing the storage with invalid quota value
  - ◆ 3.3 Failure when trying to change the quota to a value that is less than the current usage
  - ◆ 3.4 Success setting new InfoStore quota
- 4 See also

## 28.2 Request

PUT /accounts/{account}/email/{domain}/infostore

### 28.2.1 URI Parameters

account - *string*  
domain - *string*

### 28.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 28.2.3 Request Body

```
{  
  "quota": {newQuotaSize}  
}
```

#### 28.2.3.1 Parameters

quota- *Integer*  
The new quota size.

## 28.3 Response

### 28.3.1 Status Code

200 Ok  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

## 28.4 Examples

### 28.4.1 Failure setting new InfoStore quota with missing required parameter

#### Request

PUT /accounts/{account}/email/test.com/infostore

```
{  
  "foo": "bar"  
}
```

#### Response

400 Bad Request

```
{"badRequest": {"message": "POST data error", "code": 400, "details": {"quota": "Required"}}
```

## 28.4.2 Failure changing the storage with invalid quota value

### Request

```
PUT /accounts/{account}/email/test.com/infostore
{
  "quota": -1
}
```

### Response

```
400 Bad Request
{"badRequest": {"message": "POST data error", "code": 400, "details": {"quota": "-1 is less than minimum value 0"}}
```

## 28.4.3 Failure when trying to change the quota to a value that is less than the current usage

### Request

```
PUT /accounts/{account}/email/test.com/infostore
{
  "quota": 100
}
```

### Response

```
400 Bad Request
{"badRequest": {"message": "POST data error", "code": 400, "details": {"message": "Used storage exceeds the quota value"}}
```

## 28.4.4 Success setting setting new InfoStore quota

### Request

```
PUT /accounts/{account}/email/test.com/infostore
{"quota": 50}
```

### Response

```
204 No Content
```

## 28.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 29 Accounts-account-email-domain-statistics-mailboxTypeCount:GET



## 30 GET /accounts/{account}/email/{domain}/statistics/mailboxTypeCount

Retrieves per domain statistics about webmail usage

### 30.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/statistics/mailboxTypeCount
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
  - ◆ 1.4 See also

### 30.2 Request

GET /accounts/{account}/email/{domain}/statistics/mailboxTypeCount

#### 30.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

### 30.3 Response

#### 30.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or account does not exist.

#### 30.3.2 Response Body

```
{
  "numStandard": {numStandard},
  "numActiveSync": {numActiveSync},
  "numPremium": {numPremium},
  "numExchange2013Standard": {numExchange2013Standard},
  "numExchange2013Premium": {numExchange2013Premium}
  "links": [
    {
      "href": "{href}",
      "rel": "{rel}"
    }
  ]
}
```

##### 30.3.2.1 Parameters

numStandard - *integer*  
Number of mailboxes for the domain with standard webmail access

numActiveSync - *integer*  
Number of mailboxes for the domain with activesync webmail access

numPremium - *integer*  
Number of mailboxes for the domain with premium webmail access

numExchange2013Standard - *integer*  
Number of mailboxes for the domain with standard exchange access

numExchange2013Premium - *integer*  
Number of mailboxes for the domain with premium exchange access

href - *string*  
Link to other resources relevant to the mailboxes lists

rel - *string*  
Type of relation to the resource for the provided link

### 30.4 Examples

#### 30.4.1 Success scenario

##### Request

GET accounts/test-account123/email/test.com/statistics/mailboxTypeCount

## Response

```
{
  "numStandard": 1,
  "numActiveSync": 0,
  "numPremium": 0,
  "numExchange2013Standard": 5,
  "numExchange2013Premium": 2
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/statistics/mailboxTypeCount/",
      "rel": "self"
    }
  ]
}
```

## 30.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 31 Accounts-account-email-domain-username-mailboxName-autoresponder:GET

## 32 GET accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder

Gets the autoresponder info.

### 32.1 Contents

- 1 GET  
accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting the autoresponder info

### 32.2 Request

GET accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder

#### 32.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 32.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 32.3 Response

#### 32.3.1 Status Code

200 Ok  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 32.3.2 Response Body

```
{
  "autoresponder": {
    "body": "{body}",
    "active": {active},
    "subject": "{subject}"
  },
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder/",
      "rel": "self"
    }
  ]
}
```

##### 32.3.2.1 Parameters

body - *string*  
The body of auto-responder.

active - *boolean*  
The status of auto-responder.

subject - *string*  
The subject of auto-responder.

### 32.4 Examples

#### 32.4.1 Success getting the autoresponder info

##### Request

GET accounts/test-account123/email/test.com/usernames/test/autoresponder

## Response

```
200 OK
{
  "body": "AutoRespond",
  "active": true,
  "links": [
    {
      "href": "http://coreapi01.ote.chicago.hostway:8095/accounts/test-account123/email/test.com/usernames/test/autoresponder",
      "rel": "self"
    }
  ],
  "subject": "Subject"
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 33 Accounts-account-email-domain-username-mailboxName-autoresponder:PUT

## 34 PUT accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder

Activates/deactivates the autoresponder for specified mailbox, domain and account.

### 34.1 Contents

- 1 PUT  
accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure setting the autoresponder with missing required parameter
    - ◇ 1.3.2 Success setting the auto-responder
    - ◇ 1.3.3 Success deactivating the auto-responder

### 34.2 Request

PUT accounts/{account}/email/{domain}/usernames/{mailbox}/autoresponder

#### 34.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 34.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 34.2.3 Request Body

```
{  
  "subject": "{subject}",  
  "body": "{body}",  
  "active": {active}  
}
```

##### 34.2.3.1 Parameters

subject - *string*  
The subject of auto-responder.

body - *string*  
The body of auto-responder. Required.

active - *boolean*  
Sets the status of auto-responder. Required.

### 34.3 Response

#### 34.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

### 34.4 Examples

#### 34.4.1 Failure setting the autoresponder with missing required parameter

##### Request

PUT accounts/test-account123/email/test.com/usernames/test/autoresponder

```
{"subject": "Subject", "active": true}
```

### Response

```
400 Bad Request  
{ "body": "Required" }
```

## 34.4.2 Success setting the auto-responder

### Request

```
PUT accounts/test-account123/email/test.com/usernames/test/autoresponder  
{ "subject": "Subject", "body": "AutoRespond", "active": true }
```

### Response

```
204 No Content
```

## 34.4.3 Success deactivating the auto-responder

### Request

```
PUT accounts/test-account123/email/test.com/usernames/test/autoresponder  
{ "body": "AutoRespond", "active": false }
```

### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)



5 Accounts-account-email-domain-username-mailboxName-blackwhitelist-address:DELETE

## 36 DELETE

### /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}

Remove black/white listed address

#### 36.1 Contents

- 1 DELETE  
/accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Delete address from black/white list

#### 36.2 Request

DELETE /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}

##### 36.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*  
address - *string*

##### 36.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 36.3 Response

##### 36.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain, mailbox or address does not exist.

#### 36.4 Examples

##### 36.4.1 Delete address from black/white list

###### Request

```
DELETE /accounts/account-number/email/test.com/usernames/test/blackwhitelist/mb1-white20150114-0113@somedomain.com
```

###### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**37 Accounts-account-email-domain-username-mailboxName-blackwhitelist-address:GET**

## 38 GET

### /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}

Get black/white listed address details

#### 38.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}
- ◆ 1.1 Request
  - ◇ 1.1.1 URI Parameters
  - ◇ 1.1.2 Request Headers
- ◆ 1.2 Response
  - ◇ 1.2.1 Status Code
  - ◇ 1.2.2 Response Body
    - 1.2.2.1 Parameters
- ◆ 1.3 Examples
  - ◇ 1.3.1 Get address details

#### 38.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist/{address}

##### 38.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*  
address - *string*

##### 38.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 38.3 Response

##### 38.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or the mailbox does not exist.

##### 38.3.2 Response Body

```
{
  "type": "{type}",
  "address": "{address}",
  "links": {links}
}
```

###### 38.3.2.1 Parameters

type - *string*  
Type of address included in black/white list. Possible values are "W" or "B". "W" means the mailbox is included in "whitelist", "B" - included in "blacklist".

address - *string*  
The address of the black/whitelisted mailbox or domain.

links - *list*  
[Hypermedia](#) to the get black/whitelisted address

#### 38.4 Examples

##### 38.4.1 Get address details

###### Request

GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist/mb1-white20150114-0113@somedomain.com

###### Response

200 OK

```
{
  "type": "W",
  "address": "mb1-white20150114-0113@somedomain.com",
  "links": [{
    "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist/mb1-white20150114-0113@somedomain.com/",
    "rel": "self"
  }]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 39 Accounts-account-email-domain-username-mailboxName-blackwhitelist:GET

## 40 GET /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist

Retrieves a list of all addresses included in black/white list for a specific mailbox

### 40.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Get all addresses included in black/white list
    - ◇ 1.3.2 Get filtered list by address
    - ◇ 1.3.3 Get filtered list by type
    - ◇ 1.3.4 Paginated request
    - ◇ 1.3.5 Sorted request

### 40.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist

#### 40.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 40.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 40.3 Response

#### 40.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or the mailbox does not exist.

#### 40.3.2 Response Body

```
{
  "totalCount": {totalCount},
  "list": [
    {
      "type": "{type}",
      "address": "{address}"
    }
  ],
  "links": [
    {
      "href": "{href}",
      "rel": "{rel}"
    }
  ]
}
```

##### 40.3.2.1 Parameters

totalCount - *int*  
The total count of addresses included in black/white list

type - *string*  
Type of address included in black/white list. Possible values are "W" or "B". "W" means the address is included in "whitelist", "B" - included in "blacklist".

address - *string*  
The address of the black/whitelisted mailbox or domain.

links - *list*

## 40.4 Examples

### 40.4.1 Get all addresses included in black/white list

#### Request

```
GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist
```

#### Response

200 OK

```
{
  "totalCount": 4,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-black@domain.com"
    },
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    },
    {
      "type": "B",
      "address": "@sometestdomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist",
      "rel": "self"
    }
  ]
}
```

### 40.4.2 Get filtered list by address

#### Request

```
GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist?filterAddress=*somedomain.com*
```

#### Response

200 OK

```
{
  "totalCount": 2,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist?filterAddress=*somedomain.com*",
      "rel": "self"
    }
  ]
}
```

### 40.4.3 Get filtered list by type

#### Request

```
GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist?filterType=W
```

#### Response

200 OK

```
{
  "totalCount": 1,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist?filterType=W",
      "rel": "self"
    }
  ]
}
```



```
}
```

## 40.4.4 Paginated request

### Request

```
GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist?pageSize=2&page=1
```

### Response

```
200 OK
```

```
{
  "totalCount": 2,
  "list": [
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-white@domain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist?pageSize=2&page=2",
      "rel": "next"
    },
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist?pageSize=2&page=2",
      "rel": "last"
    }
  ]
}
```

## 40.4.5 Sorted request

### Request

```
GET /accounts/account-number/email/test.com/usernames/test/blackwhitelist?sortField=address&sortOrder=desc
```

### Response

```
200 OK
```

```
{
  "totalCount": 4,
  "list": [
    {
      "type": "B",
      "address": "mb3-black@somedomain.com"
    },
    {
      "type": "B",
      "address": "mb2-black@domain.com"
    },
    {
      "type": "W",
      "address": "mb1-white@somedomain.com"
    },
    {
      "type": "B",
      "address": "@sometestdomain.com"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/account-number/email/test.com/usernames/test/blackwhitelist?sortField=address&sortOrder=desc",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**41 Accounts-account-email-domain-username-mailboxName-blackwhitelist:PUT**

## 42 PUT /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist

Adding/updating addresses to black/white list for a specific mailbox

### 42.1 Contents

- 1 PUT  
/accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure adding address with missing required parameter
    - ◇ 1.3.2 Failure adding address with not valid address and type
    - ◇ 1.3.3 Success adding mailbox addresses to blackwhitelist
    - ◇ 1.3.4 Success adding domain addresses to blackwhitelist

### 42.2 Request

PUT /accounts/{account}/email/{domain}/usernames/{mailbox}/blackwhitelist

#### 42.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 42.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 42.2.3 Request Body

```
{
  "list": [
    {
      "address": "{address}",
      "type": "{type}"
    }
  ]
}
```

##### 42.2.3.1 Parameters

address - *string*  
Must be a valid email or domain address. Required.

type - *string*  
One of W, B. Required.

### 42.3 Response

#### 42.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the address does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or the mailbox does not exist.

### 42.4 Examples

### 42.4.1 Failure adding address with missing required parameter

#### Request

```
PUT /accounts/account-number/email/test.com/usernames/test/blackwhitelist
{"list":[{"address":"mb1-white@somedomain.com"}]}
```

#### Response

```
400 Bad Request
{
  "badRequest": {
    "message": "POST data error",
    "code": 400,
    "details": {
      "list.0.type": "Required"
    }
  }
}
```

### 42.4.2 Failure adding address with not valid address and type

#### Request

```
PUT /accounts/account-number/email/test.com/usernames/test/blackwhitelist
{"list":[{"address":"somedomain.com","type":"A"}]}
```

#### Response

```
400 Bad Request
{
  "badRequest": {
    "message": "POST data error",
    "code": 400,
    "details": {
      "list.0.type": "\"A\" is not one of B, W",
      "list.0.address": "Invalid address format: somedomain.com"
    }
  }
}
```

### 42.4.3 Success adding mailbox addresses to blackwhitelist

#### Request

```
PUT /accounts/account-number/email/test.com/usernames/test/blackwhitelist
{
  "list":[
    {
      "address":"mb1-white@somedomain.com",
      "type":"W"
    },
    {
      "address":"mb2-white@somedomain.com",
      "type":"W"
    }
  ]
}
```

#### Response

```
204 No Content
```

### 42.4.4 Success adding domain addresses to blackwhitelist

#### Request

```
PUT /accounts/account-number/email/test.com/usernames/test/blackwhitelist
{
  "list": [
    {
      "type": "W",
      "address": "@domainaddr1.com"
    },
    {
      "type": "B",
      "address": "@someotherdomain.com"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 43 Accounts-account-email-domain-username-mailboxName-cleanmailplus:GET

## 44 GET accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus

Gets mailbox's CleanMailPlus info.

### 44.1 Contents

- 1 GET  
accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure getting CleanMailPlus info for a non-existent domain
    - ◇ 1.3.2 Success getting CleanMailPlus info

### 44.2 Request

GET accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus

#### 44.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 44.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 44.3 Response

#### 44.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or mailbox does not exist.

#### 44.3.2 Response Body

```
[  
  {  
    "selected": "{selected_option_value}",  
    "fieldName": "{field_name}",  
    "description": "{description}",  
    "options": [  
      {  
        "name": "{option_name}",  
        "value": "{option_value}"  
      }  
    ]  
  }  
]
```

##### 44.3.2.1 Parameters

selected\_option\_value - *string*  
Currently selected option value.  
field\_name - *string*  
Field name.  
description - *string*  
Descriptive option info.  
options - *list*  
List of options.  
options.name - *string*  
Option's name.  
options.value - *string*  
Option's value.

## 44.4 Examples

### 44.4.1 Failure getting CleanMailPlus info for a non-existent domain

#### Request

```
GET accounts/test-account123/email/non-existent-domain.com/usernames/test/cleanmailplus
```

#### Response

```
404 Not Found
```

### 44.4.2 Success getting CleanMailPlus info

#### Request

```
GET accounts/test-account123/email/test.com/usernames/test/cleanmailplus
```

#### Response

```
200 OK
```

```
[
  {
    "selected": "6",
    "fieldName": "SPAM_LEVEL",
    "description": "SPAM Tag Level",
    "options": [
      {
        "name": "Least Aggressive",
        "value": "6"
      },
      {
        "name": "Less Aggressive",
        "value": "5"
      },
      {
        "name": "Normal",
        "value": "4"
      },
      {
        "name": "More Aggressive",
        "value": "3"
      },
      {
        "name": "Most Aggressive",
        "value": "2"
      },
      {
        "name": "Ultra Aggressive",
        "value": "1"
      }
    ]
  },
  {
    "selected": "2",
    "fieldName": "SPAM_ACTION",
    "description": "SPAM Detected Action",
    "options": [
      {
        "name": "None",
        "value": "0"
      },
      {
        "name": "Tag Subject",
        "value": "2"
      },
      {
        "name": "Move to SPAM Folder",
        "value": "1"
      },
      {
        "name": "Delete",
        "value": "3"
      }
    ]
  },
  {
    "selected": "1",
    "fieldName": "ATTACHMENT_CONTROL",
    "description": "Attachment Control",
    "options": [
      {
        "name": "Enable",
        "value": "1"
      },
      {
        "name": "Disable",
        "value": "0"
      }
    ]
  },
  {
    "selected": "1",
    "fieldName": "VIRUS_ACTION",
    "description": "Virus Detected Action",
    "options": [
      {
        "name": "No Virus Filter",
        "value": "0"
      }
    ]
  }
]
```

```

    },
    {
      "name": "Delete Automatically",
      "value": "1"
    }
  ]
},
{
  "selected": "0",
  "fieldName": "SPAM_LEARNING",
  "description": "SPAM Learning Status",
  "options": []
},
{
  "selected": "20",
  "fieldName": "SPAM_DISCARD_LEVEL",
  "description": "SPAM Discard Level",
  "options": []
},
{
  "selected": "0",
  "fieldName": "PHISHING_CHECKS",
  "description": "Phishing Checks",
  "options": [
    {
      "name": "Enabled",
      "value": "1"
    },
    {
      "name": "Disabled",
      "value": "0"
    }
  ]
},
{
  "selected": "0",
  "fieldName": "CLEANMAIL_PLUS",
  "description": "CleanMail Plus Enable",
  "options": [
    {
      "name": "Enable",
      "value": "1"
    },
    {
      "name": "Disable",
      "value": "0"
    }
  ]
},
{
  "selected": "1",
  "fieldName": "SPAM_CHECK",
  "description": "SPAM Check Option",
  "options": [
    {
      "name": "Enable",
      "value": "1"
    },
    {
      "name": "Disable",
      "value": "0"
    }
  ]
}
]
}
]

```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)



**45 Accounts-account-email-domain-username-mailboxName-cleanmailplus:PUT**

## 46 PUT accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus

Updates mailbox's CleanMailPlus info.

### 46.1 Contents

- 1 PUT accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating the CleanMailPlus filter attributes for a non-existent domain
    - ◇ 1.3.2 Failure updating the CleanMailPlus filter attributes by providing bad request body
    - ◇ 1.3.3 Failure updating the cleanmailplus filter attribute by providing a non-existent attribute name
    - ◇ 1.3.4 Failure updating the cleanmailplus filter attribute by providing a non-valid attribute value
    - ◇ 1.3.5 Success updating the CleanMailPlus filter attributes

### 46.2 Request

PUT accounts/{account}/email/{domain}/usernames/{mailbox}/cleanmailplus

#### 46.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 46.2.2 Request Headers

Authorization - *HTTP Authorization header [1]*  
The **Authentication** credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 46.2.3 Request Body

```
{
  "attributes": [
    {
      "attributeName": "{attribute_name}",
      "attributeValue": "{attribute_value}"
    }
  ]
}
```

##### 46.2.3.1 Parameters

attributeName - *string*  
Attribute's name.  
attributeValue - *string*  
New attribute's value.

### 46.3 Response

#### 46.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or mailbox does not exist.

### 46.4 Examples

#### 46.4.1 Failure updating the CleanMailPlus filter attributes for a non-existent domain

Request

```
PUT accounts/{account}/email/non-existent-domain.com/usernames/mailbox-cleanmailplus/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "1"
    }
  ]
}
```

### Response

```
404 Not Found
{
  "itemNotFound": {
    "message": "Resource not found",
    "code": 404,
    "details": {
      "message": "Domain not found"
    }
  }
}
```

## 46.4.2 Failure updating the CleanMailPlus filter attributes by providing bad request body

### Request

```
PUT accounts/{account}/email/test.com/usernames/mailbox-cleanmailplus/cleanmailplus
{
  "badBody": "badBodyValue"
}
```

### Response

```
400 Bad Request
{
  "attributes": "Required"
}
```

## 46.4.3 Failure updating the cleanmailplus filter attribute by providing a non-existent attribute name

### Request

```
PUT accounts/{account}/email/test.com/usernames/mailbox-cleanmailplus/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "NON_EXISTENT_ATTRIBUTE_NAME",
      "attributeValue": "1"
    }
  ]
}
```

### Response

```
400 Bad Request
{
  "attributes.0": "NON_EXISTENT_ATTRIBUTE_NAME is not valid filter attribute name"
}
```

## 46.4.4 Failure updating the cleanmailplus filter attribute by providing a non-valid attribute value

### Request

```
PUT accounts/{account}/email/test.com/usernames/mailbox-cleanmailplus/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "val01"
    }
  ]
}
```

### Response

```
400 Bad Request
{
  "attributes.0": "val01 is not valid attribute option for SPAM_LEVEL filter attribute"
}
```

## 46.4.5 Success updating the CleanMailPlus filter attributes

### Request

```
PUT accounts/{account}/email/test.com/usernames/test/cleanmailplus
{
  "attributes": [
    {
      "attributeName": "SPAM_LEVEL",
      "attributeValue": "1"
    },
  ],
}
```

```
{
  "attributeName": "SPAM_ACTION",
  "attributeValue": "1"
}
]
```

## Response

204 No Content

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 47 Accounts-account-email-domain-username-mailboxName-exchange-action:POST

## 48 POST

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/action

Apply an action on exchange mailbox e.g. **reset password**

#### 48.1 Contents

- 1 POST  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/action
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
  - ◆ 1.2 Response
    - ◇ 1.2.1 Expected Response Codes
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Request password reset of exchange mailbox
    - ◇ 1.3.2 Request password reset of exchange mailbox with wrong answer
  - ◆ 1.4 See also

#### 48.2 Request

POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/action

##### 48.2.1 Request Parameters

account - *string*  
The user account owning the domain and the exchange mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the exchange mailbox.

##### 48.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

##### 48.2.3 Request Body

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk": "{answer}"},
    "password": "{password}"
  }
}
```

action - *string*  
The action that should be performed. Allowed values are:  
**resetPassword** - Reset the password of exchange mailbox.

actionData - *dictionary (Required)*  
Data to be send with the action.

secretQuestionsAnswers - *dictionary (Required)*  
Dictionary of key/value pairs, each of which represents **secret question text key/answer** pair.

tk - *string (Required)*  
The text key for the secret question. There should be no duplicate text keys per exchange mailbox.

answer - *string (Required)*  
The answer in plain-text - must be at least two characters.

password - *string (Required)*  
The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.  
Password validation rules:

- Password must not be used more than once
- Minimum length 8
- Maximum length 128
- Should have at least one digit
- Should have at least one capital letter
- Should have at least one lower case letter
- Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)
- Should not start with the phrase 'pass' case-insensitively
- Should not start with 'abc' case-insensitively
- Should not start with 123
- Must not contain second-level domain, username or start with the first 3 letters from the username

## 48.3 Response

### 48.3.1 Expected Response Codes

204 No Content  
Success

400 Bad Request  
The supplied action or the supplied data is invalid.

404 Not Found  
The resource does not exist.

409 Conflict  
Validation of the secret answer failed.

## 48.4 Examples

### 48.4.1 Request password reset of exchange mailbox

#### Request

```
POST /accounts/account123/email/test.com/usernames/mailbox123/exchange/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk1": "answer123"},
    "password": "pWord123$"
  }
}
```

#### Response

204 No Content

### 48.4.2 Request password reset of exchange mailbox with wrong answer

#### Request

```
POST /accounts/account123/email/test.com/usernames/mailbox123/exchange/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswersList": {"tk1": "wrongAnswer"},
    "password": "pWord123$"
  }
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "message": "Invalid secret questions and answers provided",
    "code": 409,
  }
}
```

## 48.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)





## 50 GET

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats

Retrieves exchange message count for INBOX, SENT and SPAM folders

#### 50.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success
  - ◆ 1.4 See also

#### 50.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats

##### 50.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

##### 50.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

#### 50.3 Response

##### 50.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or the mailbox does not exist.  
Unable to get message count for virtual mailboxes.

500 Internal Server Error  
Missing Domain or Installation records

502 Bad Gateway  
Unable to get message count. No IMAP connection  
Unable to get message count. IMAP search operation failed.  
Unable to get message count. IMAP folder selection failed.

##### 50.3.2 Response Body

```
{
  "messageCount" : {
    "inbox" : {
      "allMessages" : {inbox_all_messages_count},
      "unreadMessages" : {inbox_unread_messages_count}
    },
    "sent" : {
      "allMessages" : {sent_all_messages_count},
      "unreadMessages" : {sent_unread_messages_count}
    },
    "spam" : {
      "allMessages" : {spam_all_messages_count},
      "unreadMessages" : {sent_unread_messages_count}
    }
  },
  "links": [{"href": "https://api.hostway.com/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats/", "rel": "self"}
}
```

###### 50.3.2.1 Parameters

inbox\_all\_messages\_count- *integer*  
All messages for INBOX folder

inbox\_unread\_messages\_count- *integer*  
Unread messages for INBOX folder

sent\_all\_messages\_count- *integer*  
All messages for SENT folder

sent\_unread\_messages\_count- *integer*  
Unread messages for SENT folder

spam\_all\_messages\_count- *integer*  
All messages for SPAM folder

spam\_unread\_messages\_count- *integer*  
Unread messages for SPAM folder

## 50.4 Examples

### 50.4.1 Success

#### Request

```
GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats
```

#### Response

200 OK

```
{
  "messageCount" : {
    "inbox" : {
      "allMessages" : 3,
      "unreadMessages" : 3
    },
    "sent" : {
      "allMessages" : 0,
      "unreadMessages" : 0
    },
    "spam" : {
      "allMessages" : 0,
      "unreadMessages" : 0
    }
  },
  "links": [{"href": "https://api.hostway.com/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/messagestats/", "rel": "self"}
}
```

## 50.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**51 Accounts-account-email-domain-username-mailboxName-exchange:DELETE**

## 52 DELETE

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange

Deletes exchange mailbox.

#### 52.1 Contents

- 1 DELETE  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success deleting exchange

#### 52.2 Request

DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange

##### 52.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

##### 52.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The **Authentication** credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 52.3 Response

##### 52.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or mailbox do not exist.

409 Conflict  
The domain is disabled.

#### 52.4 Examples

##### 52.4.1 Success deleting exchange

###### Request

DELETE /accounts/test-account/email/test.com/usernames/test/exchange

###### Response

204 No Content

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 53 Accounts-account-email-domain-username-mailboxName-exchange:GET

## 54 GET accounts/{account}/email/{domain}/usernames/{mailbox}/exchange

Gets Exchange mailbox detailed info.

### 54.1 Contents

- 1 GET accounts/{account}/email/{domain}/usernames/{mailbox}/exchange
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure getting Exchange mailbox info for a non-existent domain
    - ◇ 1.3.2 Success getting Exchange mailbox info

### 54.2 Request

GET accounts/{account}/email/{domain}/usernames/{mailbox}/exchange

#### 54.2.1 URI Parameters

account - *string*

domain - *string*

mailbox - *string*

The exchange mailbox name. String should meet the regexp [a-zA-Z0-9.\_]+, max 64 characters

#### 54.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The *Authentication* credentials of the client application.

Content-Type

Required. Set this header to application/json; charset=UTF-8

### 54.3 Response

#### 54.3.1 Status Code

200 OK

Success

401 Unauthorized

The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found

The domain name or mailbox does not exist.

#### 54.3.2 Response Body

```
{
  "maxReceiveSize": "{maxReceiveSize}",
  "maxSendSize": "{maxSendSize}",
  "primarySmtpAddress": "{primarySmtpAddress}",
  "quota": "{quota}",
  "recipientLimits": "{recipientLimits}",
  "rejectMessagesFromSendersOrMembers": "{rejectMessagesFromSendersOrMembers}",
  "requireSenderAuthenticationEnabled": false,
  "totalItemSize": "{totalItemSize}",
  "activeSyncAllowedDeviceIDs": [""],
  "activeSyncBlockedDeviceIDs": [""],
  "activeSyncEnabled": true,
  "activeSyncMailboxPolicy": "Default",
  "hasActiveSyncDevicePartnership": false,
  "imapEnabled": true,
  "mapiEnabled": true,
  "owaEnabled": true,
  "popEnabled": true,
  "serverName": "{serverName}",
  "whenChanged": "{whenChanged}",
  "whenCreated": "{whenCreated}",
  "city": "{city}",
  "company": "{company}",
  "country": "{country}",
  "department": "{department}",
  "emailAddress": "{emailAddress}",
  "fax": "{fax}",
  "homePhone": "{homePhone}",
  "manager": "{manager}",
  "mobilePhone": "{mobilePhone}",
  "notes": "{notes}",
  "office": "{office}",
  "postalCode": "{postalCode}",
  "recipientType": "{recipientType}",
```

```

"resourceType": "{resourceType}",
"type": "{type}",
"state": "{state}",
"street": "{street}",
"title": "{title}",
"webPage": "{webPage}",
"workPhone": "{workPhone}",
"displayName": "{displayName}",
"firstName": "{firstName}",
"lastName": "{lastName}",
"username": "{username}",
"domain": "{domain}",
"alias": "{alias}"
"acceptMessagesOnlyFromSendersOrMembers": "{acceptMessagesOnlyFromSendersOrMembers}",
"archiveDatabase": "{archiveDatabase}",
"archiveQuota": "{archiveQuota}",
"database": "{database}",
"deliverToMailboxAndForward": false,
"emailAddresses": [
  {
    "address": "{address}",
    "header": "smtp"
  },
  {
    "address": "{address}",
    "header": "SMTP"
  }
],
"forwardingSmtAddress": "{forwardingSmtAddress}",
"hiddenFromAddressListsEnabled": {hiddenFromAddressListsEnabled},
"itemCount": "{itemCount}",
"links": [
  {
    "href": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailbox}/exchange/",
    "rel": "self"
  },
  {
    "href": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailbox}/secrets",
    "rel": "related"
  }
]
}

```

### 54.3.2.1 Parameters

**maxReceiveSize** - *string*

The MaxReceiveSize parameter specifies the maximum size of messages that this mailbox can receive. You must specify either an integer or unlimited.

**maxSendSize** - *string*

The MaxSendSize parameter specifies the maximum size of messages that this mailbox can send. You must specify either an integer or unlimited.

**primarySmtAddress** - *string*

primary SMTP address

**quota** - *string*

Quota

**recipientLimits** - *string*

The RecipientLimits parameter specifies the maximum number of recipients per message to which this mailbox can send. You must specify either an integer or unlimited.

**rejectMessagesFromSendersOrMembers** - *list*

The RejectMessagesFromSendersOrMembers parameter specifies the identity of recipients from whom messages are rejected.

**requireSenderAuthenticationEnabled** - *boolean*

The RequireSenderAuthenticationEnabled parameter specifies whether senders must be authenticated. The two possible values for this parameter are true or false.

**totalItemSize** - *string*

total mailbox usage

**activeSyncAllowedDeviceIDs** - *list*

Active Sync allowed devices.

**activeSyncBlockedDeviceIDs** - *list*

blocked active sync devices.

**activeSyncEnabled** - *boolean*

allow mailbox access though active sync.

**activeSyncMailboxPolicy** - *string*

active sync mailbox policy name

**hasActiveSyncDevicePartnership** - *boolean*

return active sync device partnership exists of not

**imapEnabled** - *boolean*

allow mailbox access though imap.

**mapiEnabled** - *boolean*

allow mailbox access though mapi (Outlook).

**owaEnabled** - *boolean*

allow mailbox access though owa (Web mail).

**popEnabled** - *boolean*

allow mailbox access though pop3.

**serverName** - *string*

server name where user mailbox is located.

**whenChanged** - *string*

mailbox changed date.

**whenCreated** - *string*

mailbox created date.

**city** - *string*

City Name

**company** - *string*

Company Name

**country** - *string*

Country Name  
 department - *string*  
 Department Name  
 emailAddress - *string*  
 Email address, this is not actually SMTP proxy address.  
 fax - *string*  
 FAX number  
 homePhone - *string*  
 Home phone number  
 manager - *string*  
 manager  
 mobilePhone - *string*  
 Mobile phone number  
 notes - *string*  
 Notes  
 office - *string*  
 Office Name  
 postalCode - *string*  
 Zip code  
 recipientType - *string*  
 resourceType - *string*  
 The resourceType parameter specifies the type of recipients returned. Recipient types are divided into recipient types and subtypes. Each recipient type contains all common properties for all subtypes. For example, the type UserMailbox represents a user account in Active Directory that has an associated mailbox. Because there are several mailbox types, each mailbox type is identified by the RecipientTypeDetails parameter. For example, a conference room mailbox has RecipientTypeDetails set to RoomMailbox, whereas a user mailbox has RecipientTypeDetails set to UserMailbox.

This parameter returns the following values

- RoomMailbox
- EquipmentMailbox
- LegacyMailbox
- LinkedMailbox
- UserMailbox
- DiscoveryMailbox
- SharedMailbox

type - *string*  
 the Exchange account type.

This parameter returns the following values

- standard
- premium

state - *string*  
 State / Province  
 street - *string*  
 Street name  
 title - *string*  
 Title  
 webPage - *string*  
 web URL  
 workPhone - *string*  
 work phone number  
 displayName - *string*  
 Display Name  
 firstName - *string*  
 First name  
 lastName - *string*  
 Last name  
 username - *string*  
 Username  
 domain - *string*  
 Domain Name which mailbox belong to  
 alias - *string*  
 Alias, actually this is equal to username.  
 acceptMessagesOnlyFromSendersOrMembers - *string*  
 The AcceptMessagesOnlyFromSendersOrMembers parameter specifies the recipients who can send e-mail messages to this mailbox. You can specify users, contacts, or distribution groups. If you specify a distribution group, messages are accepted from all recipients that are members of that distribution group. You can also specify Exchange as a valid recipient for this parameter. If you configure a distribution group to accept messages only from the Exchange recipient, the distribution group only receives system-generated messages  
 archiveDatabase - *string*  
 Archive Database name  
 archiveQuota - *string*  
 Archive Quota  
 database - *string*  
 Mailbox database name  
 deliverToMailboxAndForward - *boolean*  
 The DeliverToMailboxAndForward parameter specifies whether messages sent to this mailbox are forwarded to another address. If the DeliverToMailboxAndForward parameter is set to true, messages are delivered to the mailbox and to the forwarding address. If set to false, messages are delivered only to the forwarding address.  
 emailAddresses - *list*  
 all SMTP proxy address.  
 address - *string*  
 SMTP proxy address  
 header - *string*  
 protocol. if HEAD is capital like SMTP, it is primary SMTP address. smtp is secondary address.

forwardingSmtAddress - *string*



The `ForwardingSmtpAddress` parameter specifies a forwarding SMTP address.

`hiddenFromAddressListsEnabled` - *boolean*  
 The `HiddenFromAddressListsEnabled` parameter specifies whether this mailbox is hidden from other address lists. The two possible values for this parameter are true or false.

`itemCount` - *string*  
 Count of mailbox items

`links` - *list*  
[Hypermedia](#) to the get exchange mailbox details resources

## 54.4 Examples

### 54.4.1 Failure getting Exchange mailbox info for a non-existent domain

#### Request

```
GET accounts/test-account-id/email/non-existent-domain.com/usernames/test-mailbox/exchange
```

#### Response

```
404 Not Found
```

### 54.4.2 Success getting Exchange mailbox info

#### Request

```
GET accounts/test-account-id/email/test.com/usernames/test-mb/exchange
```

#### Response

```
200 OK
```

```
{
  "maxReceiveSize": "50 MB (52,428,800 bytes)",
  "maxSendSize": "50 MB (52,428,800 bytes)",
  "primarySmtpAddress": "test-mb@test.com",
  "quota": "1000 MB (1,048,576,000 bytes)",
  "recipientLimits": "100",
  "rejectMessagesFromSendersOrMembers": "",
  "requireSenderAuthenticationEnabled": false,
  "totalItemSize": "0",
  "activeSyncAllowedDeviceIDs": [],
  "activeSyncBlockedDeviceIDs": [],
  "activeSyncEnabled": true,
  "activeSyncMailboxPolicy": "Default",
  "hasActiveSyncDevicePartnership": false,
  "imapEnabled": true,
  "mapiEnabled": true,
  "owaEnabled": true,
  "popEnabled": true,
  "serverName": "uc4-exmbx01",
  "whenChanged": "10/11/2013 9:43:48 AM",
  "whenCreated": "10/11/2013 9:43:39 AM",
  "city": "My City",
  "company": "My Company",
  "country": "Canada",
  "department": "HR",
  "emailAddress": "test-mb@test.com",
  "fax": "111-111-1111",
  "homePhone": "222-222-2222",
  "manager": "",
  "mobilePhone": "333-333-3333",
  "notes": "Hello Notes",
  "office": "My Office",
  "postalCode": "V6C 3T3",
  "recipientType": "UserMailbox",
  "resourceType": "user",
  "type": "standard",
  "state": "BC",
  "street": "Burrard",
  "title": "Title",
  "webPage": "www.test.com",
  "workPhone": "222-222-2222",
  "displayName": "My DisplayName",
  "firstName": "My FirstName",
  "lastName": "My LastName",
  "username": "test-mb",
  "domain": "test.com",
  "alias": "test-mb-alias",
  "acceptMessagesOnlyFromSendersOrMembers": "",
  "archiveDatabase": "",
  "archiveQuota": "50 GB (53,687,091,200 bytes)",
  "database": "DAG01-DB01",
  "deliverToMailboxAndForward": false,
  "emailAddresses": [
    {
      "address": "test-mb@test.com",
      "header": "SMTP"
    },
    {
      "address": "test-mb@second.test.com",
      "header": "smtp"
    }
  ],
  "forwardingSmtpAddress": "smtp:test@google.com",
  "hiddenFromAddressListsEnabled": false,
  "itemCount": "0",
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account-id/email/test.com/usernames/test-mb/exchange/",
      "rel": "self"
    }
  ]
}
```

```
    },  
    {  
      "href": "{{APIBaseURL}}/accounts/test-account-id/email/test.com/usernames/test-mb/secrets",  
      "rel": "related"  
    }  
  ]  
}
```

**55 Accounts-account-email-domain-username-mailboxName-exchange:PUT**

## 56 PUT accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange

Creates or updates an exchange mailbox for specified account and domain.

### 56.1 Request

PUT accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange

#### 56.1.1 URI Parameters

account - *string*

domain - *string*

mailboxName - *string*

The exchange mailbox. String should meet the regexp `[a-zA-Z0-9._-]+`, max 64 characters

#### 56.1.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The [Authentication](#) credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

#### 56.1.3 Request Body

```
{
  "password": "{password}",
  "resetPassword": {resetPassword},
  "displayName": "{displayName}",
  "type": {type},
  "resourceType": "{resourceType}",
  "firstName": {firstName},
  "lastName": {lastName},
  "city": {city},
  "company": {company},
  "country": {country},
  "department": {department},
  "fax": {fax},
  "homePhone": {homePhone},
  "mobilePhone": {mobilePhone},
  "postalCode": {postalCode},
  "state": {state},
  "street": {street},
  "title": {title},
}
```

##### 56.1.3.1 Parameters

password - *string* (Required when creating mailbox)

The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.

Password validation rules:

Password must not be used more than once

Minimum length 8

Maximum length 128

Should start with a letter

Should have at least one digit

Should have at least one capital letter

Should have at least one lower case letter

Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)

Should not start with the phrase 'pass' case-insensitively

Should not start with 'abc' case-insensitively

Should not start with 123

Must not contain second-level domain, username or start with the first 3 letters from the username

resetPassword - *boolean* (Optional)

The resetPassword flag is set for "exchange" mailbox, forcing the user to change his password on next logon.

displayName - *string* (Required when creating mailbox)

Display Name(256 characters), The displayName parameter specifies the user's display name.

type - *string* - one of "standard" or "premium"

Exchange account type. Valid values are standard, premium. If not provided - standard is default.

resourceType - *string* - one of "user", "equipment" or "room"

Exchange resource type. If not provided - user is default. Passing this parameter when updating mailbox will result in 400 Bad Request error

firstName

First name(64 characters), The firstName parameter specifies the user's given name.

lastName

Last name(64 characters), The lastName parameter specifies the user's surname

city City Name(128 characters), The city parameter specifies the user's city.

company Company Name(64 characters), The company parameter specifies the user's company.

country Country Name(128 characters), The country parameter specifies the user's country or region.

department Department Name(64 characters), The department parameter specifies the user's department.

fax FAX number(64 characters), The fax parameter specifies the user's fax number.

homePhone Home phone number(64 characters), The homePhone parameter specifies the user's home telephone number.

mobilePhone Mobile phone number(64 characters), The mobilePhone parameter specifies the user's mobile phone number.

postalCode Zip code(40 characters), The postalCode parameter specifies the user's zip code or postal code.

state State / Province(128 characters), The state parameter specifies the user's state or province.

street Street name(1024 characters), The street parameter specifies the user's physical address.

title Title(128 characters), The Title parameter specifies the user's title.

## 56.2 Response

### 56.2.1 Status Code

201 Created  
Success

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

409 Conflict  
The mailbox already exists(on create) or domain is disabled(on update).

## 56.3 Examples

### 56.3.1 Trying to create exchange mailbox without a password

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"displayName": "John" }
```

#### Response

```
400 Bad Request
{"password": "Required" }
```

### 56.3.2 Trying to create exchange mailbox without a displayName

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"password": "pWord123$" }
```

#### Response

```
400 Bad Request
{"displayName": "Required" }
```

### 56.3.3 Trying to create exchange mailbox with a not valid password

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"password": "abc1234", "displayName": "John"}
```

#### Response

```
400 Bad Request
{"password":"Password should not start with 'abc' case-insensitively"}
```

### 56.3.4 Create exchange mailbox

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"password": "pWord123$", "displayName": "John"}
```

#### Response

```
201 Created
```

### 56.3.5 Create premium exchange mailbox

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"password": "pWord123$", "displayName": "John", "type": "premium"}
```

#### Response

```
201 Created
```

### 56.3.6 Update exchange mailbox password

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"password": "!q2w3e4R"}
```

#### Response

```
204 No Content
```

### 56.3.7 Create exchange room

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith.room/exchange
{"password": "pWord123$", "displayName": "John", "resourceType": "room"}
```

#### Response

```
201 Created
```

### 56.3.8 Trying to update exchange resource type

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith.rooh/exchange
{"resourceType": "equipment"}
```

#### Response

```
400 Bad Request
resourceType cannot be updated
```

### 56.3.9 Update exchange mailbox, forcing the user to reset his password on next logon

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/exchange
{"resetPassword": true}
```

## Response

204 No Content

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

7 Accounts-account-email-domain-username-mailboxName-forwards-fwdmailbox:DELETE



## 58 DELETE

### accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/{fwd-mailbox}

Deletes forwarding for specified mailbox, domain and account.

#### 58.1 Contents

- 1 DELETE  
accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/{fwd-mailbox}
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success deleting mailbox forwarding
  - ◆ 1.4 See also

#### 58.2 Request

DELETE accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/{fwd-mailbox}

##### 58.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*  
fwd-mailbox - *string*  
Target mailbox forwarding to be deleted.

##### 58.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 58.3 Response

##### 58.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 58.4 Examples

##### 58.4.1 Success deleting mailbox forwarding

###### Request

```
DELETE accounts/{account}/email/test.com/usernames/test/forwards/test-fwd-mailbox
```

###### Response

```
204 No Content
```

#### 58.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 59 Accounts-account-email-domain-username-mailboxName-forwards:GET

## 60 GET accounts/{account}/email/{domain}/usernames/{mailbox}/forwards

Gets forwarding for specified mailbox, domain and account.

### 60.1 Contents

- 1 GET  
accounts/{account}/email/{domain}/usernames/{mailbox}/forwards
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting forwarding info
  - ◆ 1.4 See also

### 60.2 Request

GET accounts/{account}/email/{domain}/usernames/{mailbox}/forwards

#### 60.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 60.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 60.3 Response

#### 60.3.1 Status Code

200 Ok  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 60.3.2 Response Body

```
{
  "targets": [
    {
      "target": "{target_mailbox}"
    },
    {
      "target": "{target_mailbox}"
    }
  ],
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/",
      "rel": "self"
    }
  ]
}
```

##### 60.3.2.1 Parameters

target\_mailbox - *string*  
The returned forward target mailbox.

### 60.4 Examples

#### 60.4.1 Success getting forwarding info

##### Request

GET accounts/{account}/email/test.com/usernames/test/forwards

##### Response

200 OK

```
<nowiki>
{
  "targets": [
    {
      "target": "mbl@test.com"
    },
    {
      "target": "test@test-forward.com"
    }
  ],
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailbox}/forwards/",
      "rel": "self"
    }
  ]
}
</nowiki>
```

## 60.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 61 Accounts-account-email-domain-username-mailboxName-forwards:POST

## 62 POST accounts/{account}/email/{domain}/usernames/{mailbox}/forwards

Sets forwarding for specified mailbox, domain and account.

### 62.1 Contents

- 1 POST accounts/{account}/email/{domain}/usernames/{mailbox}/forwards
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure when creating a forwarding with missing required parameter
    - ◇ 1.3.2 Failure when creating a forwarding with an empty list of targets
    - ◇ 1.3.3 Success when creating a forwarding
    - ◇ 1.3.4 Success setting multiple forwarding targets (distribution list)
    - ◇ 1.3.5 Creating local forwarding or distribution list that contains mailboxes in the source domain
    - ◇ 1.3.6 Failure when creating forwarding to the same mailbox (avoids forwarding loops)
    - ◇ 1.3.7 Failure when creating forwarding to same mailbox on a domain alias
    - ◇ 1.3.8 Failure when creating local forwarding to non-existing mailbox
    - ◇ 1.3.9 Failure when creating a distribution list with a missing local target mailbox
    - ◇ 1.3.10 Success creating forwarding to an external mailbox
    - ◇ 1.3.11 Success creating forwarding to multiple external mailboxes (distribution list)
  - ◆ 1.4 See also

### 62.2 Request

POST accounts/{account}/email/{domain}/usernames/{mailbox}/forwards

#### 62.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

#### 62.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 62.2.3 Request Body

```
{
  "targets": [
    { "target": "{target-email}" },
    { "target": "{target-email1}" },
  ]
}
```

##### 62.2.3.1 Parameters

targets - *list*  
List of target objects.

target - *string*  
The target mailbox to forward to.

### 62.3 Response

#### 62.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized

The supplied credentials are invalid or do not provide permissions for this operation.

#### 404 Not Found

The domain name does not exist.

#### 409 Conflict

Create of the forwarder would produce an impossible to delivery loops namely:

1. the usernames on the forward source and target are the same, **and**:
  1. the domains on the source is an alias of the target's domain name, **or**
  2. the domains on the target is an alias of the source's domain name

## 62.4 Examples

### 62.4.1 Failure when creating a forwarding with missing required parameter

#### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
{empty request body}
```

#### Response

```
400 Bad Request
{"targets": "Required"}
```

### 62.4.2 Failure when creating a forwarding with an empty list of targets

#### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
{"targets": []}
```

#### Response

```
400 Bad Request
{"targets": "Shorter than minimum length 1"}
```

### 62.4.3 Success when creating a forwarding

#### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
{
  "targets": [
    { "target": "test-fwd@example.com" }
  ]
}
```

#### Response

```
204 No Content
```

### 62.4.4 Success setting multiple forwarding targets (distribution list)

#### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
{
  "targets": [
    { "target": "test-fwd@test.com" },
    { "target": "test-fwd@example.com" }
  ]
}
```

#### Response

```
204 No Content
```

### 62.4.5 Creating local forwarding or distribution list that contains mailboxes in the source domain

Local forwarding is created when some or all of the target mailboxes are in the same domain as the source mailbox. In case there are such mailboxes certain validation rules are applied that verify the target mailbox existence and that the target mailbox is different than the source. It is important to note that **none** of the targets will be created if the validation fails for **any** of the mailboxes in the list. That is: a single non-existent local target mailbox in the list will prevent all of the targets from being added regardless of the invalid target position in the list.

### 62.4.6 Failure when creating forwarding to the same mailbox (avoids forwarding loops)

#### Request

```
POST accounts/source_account/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    { "target": "test@test.com" }
  ]
}
```

### Response

400 Bad Request

```
{"test@test.com": "Is the same as current mailbox"}
```

## 62.4.7 Failure when creating forwarding to same mailbox on a domain alias

### Request

```
POST accounts/source_account/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    { "target": "test@test-com-alias.com" }
  ]
}
```

### Response

409 Conflict

```
{
  "conflict": {
    "message": "Circular target test@test-com-alias.com",
    "code": 409,
    "details": ""
  }
}
```

## 62.4.8 Failure when creating local forwarding to non-existing mailbox

### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    { "target": "not-existing-mailbox@test.com" }
  ]
}
```

### Response

400 Bad Request

```
{"not-existing-mailbox@test.com": "Target not found"}
```

## 62.4.9 Failure when creating a distribution list with a missing local target mailbox

```
POST accounts/{account}/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    { "target": "existing-mailbox@test.com" },
    { "target": "not-existing-mailbox@test.com" }
  ]
}
```

### Response

400 Bad Request

```
{"not-existing-mailbox@test.com": "Target not found"}
```

## 62.4.10 Success creating forwarding to an external mailbox

Notice the `different_domain.com` in the request is external (different than `test.com`).

### Request

```
POST accounts/{account}/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    { "target": "test-fwd@different-domain.com" }
  ]
}
```

### Response

204 No Content



### 62.4.11 Success creating forwarding to multiple external mailboxes (distribution list)

#### Request

Notice that **different\_domain\_1.com** and **different\_domain\_2.com** are both different than **test.com**.

```
POST accounts/{account}/email/test.com/usernames/test/forwards
```

```
{
  "targets": [
    {"target": "test-fwd@different-domain_1.com"},
    {"target": "test-fwd1@different-domain_2.com"}
  ]
}
```

#### Response

```
204 No Content
```

## 62.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**63 Accounts-account-email-domain-username-mailboxName-mailbox-action:POST**

## 64 POST

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/action

Apply an action on a mailbox e.g. **reset password**

#### 64.1 Contents

- 1 POST  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/action
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
  - ◆ 1.2 Response
    - ◇ 1.2.1 Expected Response Codes
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Request password reset of a mailbox
    - ◇ 1.3.2 Request password reset of a mailbox with wrong answer
  - ◆ 1.4 See also

#### 64.2 Request

POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/action

##### 64.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the mailbox.

##### 64.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

##### 64.2.3 Request Body

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk": "{answer}"},
    "password": "{password}"
  }
}
```

action - *string*  
The action that should be performed. Allowed values are:  
**resetPassword** - Reset the password of a mailbox.

actionData - *dictionary (Required)*  
Data to be send with the action.

secretQuestionsAnswers - *dictionary (Required)*  
Dictionary of key/value pairs, each of which represents **secret question text key/answer** pair.

tk - *string (Required)*  
The text key for the secret question. There should be no duplicate text keys per mailbox.

answer - *string (Required)*  
The answer in plain-text - must be at least two characters.

password - *string (Required)*  
The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.  
Password validation rules:

- Password must not be used more than once
- Minimum length 8
- Maximum length 128
- Should have at least one digit
- Should have at least one capital letter
- Should have at least one lower case letter
- Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)
- Should not start with the phrase 'pass' case-insensitively
- Should not start with 'abc' case-insensitively
- Should not start with 123
- Must not contain second-level domain, username or start with the first 3 letters from the username

## 64.3 Response

### 64.3.1 Expected Response Codes

204 No Content  
Success

400 Bad Request  
The supplied action or the supplied data is invalid.

404 Not Found  
The resource does not exist.

409 Conflict  
Validation of the secret answer failed.

## 64.4 Examples

### 64.4.1 Request password reset of a mailbox

#### Request

```
POST /accounts/account123/email/test.com/usernames/mailbox123/mailbox/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk1": "answer123"},
    "password": "pWord123$"
  }
}
```

#### Response

204 No Content

### 64.4.2 Request password reset of a mailbox with wrong answer

#### Request

```
POST /accounts/account123/email/test.com/usernames/mailbox123/mailbox/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswersList": {"tk1": "wrongAnswer"},
    "password": "pWord123$"
  }
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "message": "Invalid secret questions and answers provided",
    "code": 409,
  }
}
```

## 64.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**65 Accounts-account-email-domain-username-mailboxName-mailbox:DELETE**

## 66 DELETE

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

Deletes mailbox.

#### 66.1 Contents

- 1 DELETE
  - /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox
    - ◆ 1.1 Request
      - ◇ 1.1.1 URI Parameters
      - ◇ 1.1.2 Request Headers
    - ◆ 1.2 Response
      - ◇ 1.2.1 Status Code
    - ◆ 1.3 Examples
      - ◇ 1.3.1 Success deleting mailbox

#### 66.2 Request

DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

##### 66.2.1 URI Parameters

account - *string*  
domain - *string*  
mailbox - *string*

##### 66.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 66.3 Response

##### 66.3.1 Status Code

204 No Content  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or mailbox do not exist.

#### 66.4 Examples

##### 66.4.1 Success deleting mailbox

###### Request

DELETE /accounts/test-account/email/test.com/usernames/test/mailbox

###### Response

204 No Content

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 67 Accounts-account-email-domain-username-mailboxName-mailbox:GET

## 68 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

Get mailbox details.

### 68.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting mailbox details

### 68.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

#### 68.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

#### 68.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 68.3 Response

#### 68.3.1 Status Code

- 200 OK  
Success
- 401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.
- 404 Not Found  
The domain name does not exist.

#### 68.3.2 Response Body

```
{
  "username": "{username}",
  "master": {master},
  "quota": {quota},
  "usage": {usage},
  "authToken": {authToken},
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/",
      "rel": "self"
    }
  ]
}
"sitemail_urls":
[
  {
    "url": "http://sitemaildev7.test",
    "version": "7"
  },
  {
    "url": "http://sitemaildev8.test",
    "version": "8"
  }
]
```

##### 68.3.2.1 Parameters

username - *string*  
The mailbox username.

authToken - *string*  
Authorization string for logging into OpenXchange  
Returned only if long URL, starting with "/accounts" is used



`master` - *boolean*  
Describes if this is a master account.

`quota` - *integer*  
Mailbox quota in MB.

`usage` - *integer*  
Mailbox usage in MB.

`sitemail_urls` - *list*  
List containing maps of SiteMail URLs and their versions

## 68.4 Examples

### 68.4.1 Success getting mailbox details

#### Request

```
GET /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
```

#### Response

200 OK

```
{
  "username": "test",
  "authToken": "NGA8mH65N63ZpakAIYCWZodYRRbsOQ/Uj9TcnWWaUYuHZliHZbmT2eDFQW2zV71PYmmMhUKlTx/5UMYkG4hv2ycGb0p7C2i4U8Vh1mK4/OrQQjQHLej27zfTMR+",
  "master": true,
  "quota": 10,
  "usage": 10,
  "links": [
    {
      "href": "{APIBaseURL}/accounts/test-account123/email/test.com/usernames/john.smith/mailbox/", "rel": "self"
    }
  ]
  "sitemail_urls": [
    {
      "url": "http://sitemaildev7.test",
      "version": "7"
    },
    {
      "url": "http://sitemaildev8.test",
      "version": "8"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**69 Accounts-account-email-domain-username-mailboxName-mailbox:PUT**

## 70 PUT accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

Creates or updates a mailbox for specified account and domain. The mailbox quota is determined by the E-mail domain product specification.

### 70.1 Contents

- 1 PUT  
accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Trying to create mailbox without a password
    - ◇ 1.3.2 Trying to create mailbox with a not valid password
    - ◇ 1.3.3 Trying to create mailbox with invalid quota value (less or equal to 0)
    - ◇ 1.3.4 Trying to create mailbox when mailbox qty for OX type is exceeded
    - ◇ 1.3.5 Create mailbox
    - ◇ 1.3.6 Update mailbox password
    - ◇ 1.3.7 Trying to update mailbox password with already used one
    - ◇ 1.3.8 Update mailbox name
    - ◇ 1.3.9 Update mailbox using an existing name
    - ◇ 1.3.10 Create mailbox with specified quota
    - ◇ 1.3.11 Update existing mailbox quota
  - ◆ 1.4 See also

### 70.2 Request

PUT accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox

#### 70.2.1 URI Parameters

account - *string*

domain - *string*

mailboxName - *string*

The mailbox. String should meet the regexp [a-zA-Z0-9.\_-]+

#### 70.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

The *Authentication* credentials of the client application.

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

#### 70.2.3 Request Body

```
{
  "password": "{password}",
  "master": {master},
  "quota": {quota},
  "type": {type},
  "username": {username}
}
```

##### 70.2.3.1 Parameters

password - *string* (Required)

The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.

Password validation rules:

- Password must not be used more than once

- Minimum length 8

- Maximum length 128

- Should have at least one digit

- Should have at least one capital letter

- Should have at least one lower case letter

- Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)

- Should not start with the phrase 'pass' case-insensitively

- Should not start with 'abc' case-insensitively

- Should not start with 123

- Must not contain second-level domain, username or start with the first 3 letters from the username

master - *boolean*

Describes if this is a master account.

quota - *int*

Quota value in MB. If not provided default quota for OpenXchange type is set

type - *string*

OpenXchange account type. Supported values can be obtained from the [webmailNames API endpoint](#). If not provided - standard is default. If current mailbox qty is more than the allowed value for specific OpenXchange account type, the response will be *400 Bad Request*

username - *string*

The mailbox. String should meet the regexp `[a-zA-Z0-9._-]+`  
Should be used when updating mailbox only

## 70.3 Response

### 70.3.1 Status Code

201 Created

Success

204 No Content

Success

400 Bad Request

The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized

The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found

The domain name does not exist.

409 Conflict

Mailbox already exists.

## 70.4 Examples

### 70.4.1 Trying to create mailbox without a password

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"master": false}
```

#### Response

```
400 Bad Request
{"password": "Required"}
```

### 70.4.2 Trying to create mailbox with a not valid password

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"password": "abc1234", "master": true}
```

#### Response

```
400 Bad Request
{"password": "Required"}
```

### 70.4.3 Trying to create mailbox with invalid quota value(less or equal to 0)

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"password": "test_password1", "quota": 0}
```

#### Response

```
400 Bad Request
{"computeFault": "Quota must be greater than 0"}
```

### 70.4.4 Trying to create mailbox when mailbox qty for OX type is exceeded

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"password": "test_password1", "type": "activesync"}
```

### Response

```
400 Bad Request
{"computeFault": "Number of mailboxes exceeded"}
```

## 70.4.5 Create mailbox

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"password": "pWord123$", "master": true}
```

### Response

```
201 Created
```

## 70.4.6 Update mailbox password

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"password": "!q2w3e4R"}
```

### Response

```
204 No Content
```

## 70.4.7 Trying to update mailbox password with already used one

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith2/mailbox
{"password": "pWord123$"}
```

### Response

```
400 Bad Request
```

## 70.4.8 Update mailbox name

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"username": "john.smith2"}
```

### Response

```
204 No Content
```

## 70.4.9 Update mailbox using an existing name

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith/mailbox
{"username": "jack.smith"}
```

### Response

```
400 Bad Request
```

## 70.4.10 Create mailbox with specified quota

### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith2/mailbox
{"password": "pword1234", "quota": 70}
```

## Response

201 Created

### 70.4.11 Update existing mailbox quota

#### Request

```
PUT /accounts/test-account123/email/test.com/usernames/john.smith2/mailbox
{"quota": 85}
```

#### Response

204 No Content

## 70.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 71 Accounts-account-email-domain-username-mailboxName-messagestats:GET

## 72 GET

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats

Retrieves mailbox message count for INBOX, SENT and SPAM folders

#### 72.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success
  - ◆ 1.4 See also

#### 72.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats

##### 72.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

##### 72.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

#### 72.3 Response

##### 72.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or the mailbox does not exist.  
Unable to get message count for virtual mailboxes.

500 Internal Server Error  
Missing Domain or Installation records

502 Bad Gateway  
Unable to get message count. No IMAP connection  
Unable to get message count. IMAP search operation failed.  
Unable to get message count. IMAP folder selection failed.

##### 72.3.2 Response Body

```
{
  "messageCount" : {
    "inbox" : {
      "allMessages" : {inbox_all_messages_count},
      "unreadMessages" : {inbox_unread_messages_count}
    },
    "sent" : {
      "allMessages" : {sent_all_messages_count},
      "unreadMessages" : {sent_unread_messages_count}
    },
    "spam" : {
      "allMessages" : {spam_all_messages_count},
      "unreadMessages" : {sent_unread_messages_count}
    }
  },
  "links": [{"href": "https://api.hostway.com/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats/", "rel": "self"}
}
```

##### 72.3.2.1 Parameters

inbox\_all\_messages\_count- *integer*  
All messages for INBOX folder

inbox\_unread\_messages\_count- *integer*  
Unread messages for INBOX folder



sent\_all\_messages\_count- *integer*  
All messages for SENT folder

sent\_unread\_messages\_count- *integer*  
Unread messages for SENT folder

spam\_all\_messages\_count- *integer*  
All messages for SPAM folder

spam\_unread\_messages\_count- *integer*  
Unread messages for SPAM folder

## 72.4 Examples

### 72.4.1 Success

#### Request

```
GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats
```

#### Response

200 OK

```
{
  "messageCount" : {
    "inbox" : {
      "allMessages" : 3,
      "unreadMessages" : 3
    },
    "sent" : {
      "allMessages" : 0,
      "unreadMessages" : 0
    },
    "spam" : {
      "allMessages" : 0,
      "unreadMessages" : 0
    }
  },
  "links": [{"href": "https://api.hostway.com/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats/", "rel": "self"}
}
```

## 72.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 73 Accounts-account-email-domain-username-mailboxName-migration:POST

## 74 POST `accounts/{account}/email/{domain}/usernames/{mailboxName}/migration`

Creates a support ticket for failed mailbox migration

### 74.1 Contents

- 1 POST `accounts/{account}/email/{domain}/usernames/{mailboxName}/migration`
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success creating a support ticket
  - ◆ 1.4 See also

### 74.2 Request

POST `accounts/{account}/email/{domain}/usernames/{mailboxName}/migration`

#### 74.2.1 URI Parameters

`account` - *string*  
`domain` - *string*  
`mailboxName` - *string*

#### 74.2.2 Request Headers

`Authorization` - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

`Content-Type`  
Required. Set this header to `application/json; charset=UTF-8`

#### 74.2.3 Request Body

```
{
  "subject": "{subject}",
  "body": "{body}"
}
```

##### 74.2.3.1 Parameters

`subject` - *string*  
Subject of the support ticket

`body` - *string*  
Body of the support ticket

### 74.3 Response

#### 74.3.1 Status Code

201 Created  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

### 74.4 Examples

#### 74.4.1 Success creating a support ticket

##### Request

POST `accounts/test-account123/email/test.com/usernames/john.smith/migration`

```
{
  "subject": "subject"
  "body": "body",
}
```

## Response

201 Created

## 74.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 75 Accounts-account-email-domain-username-mailboxName-secrets:GET

## 76 GET /accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets

Retrieves a list of text keys for all mailbox(alias) secret questions.

### 76.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
  - ◆ 1.4 See also

### 76.2 Request

GET /accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets

#### 76.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

maiboxName - *string*  
The name of the mailbox

#### 76.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 76.3 Response

#### 76.3.1 Status Code

200 OK  
Success

404 Not Found  
The domain name, mailbox or account does not exist.

#### 76.3.2 Response Body

```
{
  "list": [ "{tk}", "{tk}", "{tk}" ],
  "links": [
    {
      "href": "https://api.hostway.com/accounts/{account}/email/{domain}/usernames/{maiboxName}/secrets",
      "rel": "self"
    }
  ]
}
```

##### 76.3.2.1 Parameters

list - 'list'  
A list of secret questions of the account  
tk - *string* (Unique)  
Unique secret question text key.

links - 'list'  
[Hypermedia](#) to the account secret questions resource

### 76.4 Examples

#### 76.4.1 Success scenario

##### Request

GET /accounts/test-account123/email/test.com/usernames/test-mailbox/secrets

##### Response

```
{
  "list": [ "tk1", "tk2", "tk3" ],
  "links": [
    {
      "href": "https://api.hostway.com/accounts/test-account123/email/test.com/usernames/test-mailbox/secrets",
      "rel": "self"
    }
  ]
}
```

```
} ]
```

## 76.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**77 Accounts-account-email-domain-username-mailboxName-secrets:PUT**



## 78 PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets

Set or update all secret questions and answers for mailbox(alias).

### 78.1 Contents

- 1 PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Body
      - 1.1.2.1 Parameters
    - ◇ 1.1.3 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
    - ◇ 1.3.2 Set only two secret questions and answers, when the minimum required is three
  - ◆ 1.4 See also

### 78.2 Request

PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets

#### 78.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific mailbox domain

mailboxName - *string*  
The name of the mailbox

#### 78.2.2 Request Body

```
{
  "{tk}": "{answer}",
  "{tk}": "{answer}",
  ...
}
```

##### 78.2.2.1 Parameters

tk - *string* (Required)  
The text key for the secret question. There should be no duplicate text keys per mailbox.

answer - *string* (Required)  
The answer in plain-text - must be at least two characters.

#### 78.2.3 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 78.3 Response

#### 78.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the number of the secret questions and answers pairs in the request is lower then the [required minimum](#).

401 Unauthorized  
The supplied credentials are invalid.

403 Forbidden  
The authorized user does not have permissions for this operation.

404 Not Found  
The domain name, mailbox or account does not exist.

### 78.4 Examples

#### 78.4.1 Success scenario

##### Request

PUT /accounts/test-account123/email/test.com/usernames/test-mailbox/secrets

```
{
  "tk1": "answer1",
  "tk2": "answer2",
  "tk3": "answer3"
}
```

### Response

204 No Content

## 78.4.2 Set only two secret questions and answers, when the minimum required is three

### Request

PUT /accounts/test-account123/email/test.com/usernames/test-mailbox/secrets

```
{
  "tk1": "answer1",
  "tk2": "answer2"
}
```

### Response

400 Bad Request

"details": {"": "Shorter than minimum length 3"}

## 78.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 79 Accounts-account-email-domain-username-mailboxName-webmail:GET

## 80 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

Gets OpenXchange account type.

### 80.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting account type

### 80.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

#### 80.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

#### 80.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 80.3 Response

#### 80.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 80.3.2 Response Body

```
{
  "authToken": "{authenticationToken}",
  "type": "{type}",
  "allowUpgrade": {allowUpgrade},
  "resetPassword": {resetPassword},
  "theme": "{theme}",
  "language": "{language}",
  "timezone": "{timezone}",
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail/",
      "rel": "self"
    }
  ]
}
```

##### 80.3.2.1 Parameters

authenticationToken - *string*  
Authentication token value

allowUpgrade - *boolean*  
OpenXchange allowUpgrade option

type - *string*  
OpenXchange account type. Supported values can be obtained from the [webmailNames API endpoint](#).

resetPassword - *boolean*  
Returns if customer will be forced to reset mailbox password the next time he logs in to his OX account

theme - *string*  
Theme identifier for setting a specific theme when creating an OX account

language - *string*  
OpenXchange account locale of type 'en\_US'.

timezone - *string*  
OpenXchange account timezone of type 'America/Chicago'.

## 80.4 Examples

### 80.4.1 Success getting account type

#### Request

```
GET /accounts/test/email/test.com/usernames/john.smith/webmail
```

#### Response

```
200 OK
{
  "authToken": "PoiC9iyBehel3ZHWhyfa2eArKUJy+RJ5WpyN0eCBs2+Ns8SIdivhgto9zWLDnGZWlgTV9MuJ1F+//XnoKZWxjaRr2qh++437fChyxZmaHwe7g==",
  "type": "standard",
  "allowUpgrade": true,
  "resetPassword": false,
  "theme": "test_theme",
  "language": "en_US",
  "timezone": "America/Chicago",
  "links": [
    {
      "href": "{APIBaseURL}/accounts/test/email/test.com/usernames/john.smith/webmail/",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 81 Accounts-account-email-domain-username-mailboxName-webmail:POST

## 82 POST

### /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

Creates OpenXchange account.

#### 82.1 Contents

- 1 POST  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure creating account with not-valid parameter
    - ◇ 1.3.2 Failure creating account when mailbox qty for OX type is exceeded
    - ◇ 1.3.3 Success creating account

#### 82.2 Request

POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

##### 82.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

##### 82.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

##### 82.2.3 Request Body

```
{
  "type": "{type}",
  "allowUpgrade": {allowUpgrade},
  "theme": "{theme}",
  "language": "{language}",
  "timezone": "{timezone}"
}
```

##### 82.2.3.1 Parameters

allowUpgrade - *boolean*  
OpenXchange allowUpgrade option value. The default value is true.

type - *string*  
OpenXchange account type. Supported values can be obtained from the [webmailNames API endpoint](#). If not provided - standard is default. If current mailbox qty is more than allowed for specific OpenXchange account type - the response is *400 Bad Request*

theme - *string*  
Theme identifier for setting a specific theme when creating an OX account. If not provided, default theme is set.

language - *string*  
OpenXchange account locale of type 'en\_US'.

timezone - *string*  
OpenXchange account timezone of type 'America/Chicago'.

#### 82.3 Response

##### 82.3.1 Status Code

201 Created  
Success

400 Bad Request  
The format of the request body is invalid or the OpenXchange account type does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

409 Conflict  
OpenXchange account already exists.

404 Not Found  
The domain name or mailbox do not exist.

## 82.4 Examples

### 82.4.1 Failure creating account with not-valid parameter

#### Request

```
POST /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "non-valid"}
```

#### Response

```
400 Bad Request
{"type": "\"non-existing\" is not one of standard, activesync, premium"}
```

### 82.4.2 Failure creating account when mailbox qty for OX type is exceeded

#### Request

```
POST /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "activesync"}
```

#### Response

```
400 Bad Request
{"computeFault":"Number of mailboxes exceeded"}
```

### 82.4.3 Success creating account

#### Request

```
POST /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "standard", "allowUpgrade": false, "language": "en_US", "timezone": "America/Chicago"}
```

#### Response

```
201 Created
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)



**83 Accounts-account-email-domain-username-mailboxName-webmail:PUT**

## 84 PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

Upgrades/downgrades OpenXchange account type.

### 84.1 Contents

- 1 PUT  
/accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating account type with missing required parameter
    - ◇ 1.3.2 Failure updating account type with invalid parameter
    - ◇ 1.3.3 Failure using inconsistent combination of allowUpgrade and type
    - ◇ 1.3.4 Failure changing the allowUpgrade option for activesync or premium accounts
    - ◇ 1.3.5 Failure upgrading account while allowUpgrade is set to false
    - ◇ 1.3.6 Failure upgrading account when mailbox qty for OX type is exceeded
    - ◇ 1.3.7 Success updating account type
    - ◇ 1.3.8 Failure downgrading account due to current usage greater than new quota
    - ◇ 1.3.9 Success updating resetPassword flag
    - ◇ 1.3.10 Success updating timezone, language and theme

### 84.2 Request

PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail

#### 84.2.1 URI Parameters

account - *string*  
domain - *string*  
mailboxName - *string*

#### 84.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 84.2.3 Request Body

```
{  
  "type": "{oxttype}",  
  "allowUpgrade": {allowUpgrade},  
  "resetPassword": {resetPassword},  
  "theme": "{theme}",  
  "language": "{language}",  
  "timezone": "{timezone}"  
}
```

##### 84.2.3.1 Parameters

allowUpgrade - *boolean*

OpenXchange allowUpgrade option value. If upgrades are disabled by the product, it cannot be set to true.

type - *string* (*Required*)

OpenXchange account type. Supported values can be obtained from the *webmailNames API endpoint*.

The account owner(with admin-user credentials) can make changes between account types regardless of the allowUpgrade settings.

If the call is made with the end-user credentials and allowUpgrade is set to *false* providing a type different than standard results in a *400 Bad Request* response

If the call is made with the end-user credentials and type is already different than standard and allowUpgrade is set to *false* the response is *400 Bad Request*

If current usage is more than new type quota the response is *400 Bad Request*

If current mailbox qty is more than allowed for specific OpenXchange account type the response is *400 Bad Request*

If upgrades are disabled by the product, it can only be set to the current type

resetPassword - *boolean*

Forces customer to reset his password the next time he logs in to his OX account.

theme - *string*

Theme identifier for setting a specific theme when creating an OX account.

language - *string*  
OpenXchange account locale of type 'en\_US'.  
timezone - *string*  
OpenXchange account timezone of type 'America/Chicago'.

## 84.3 Response

### 84.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

## 84.4 Examples

### 84.4.1 Failure updating account type with missing required parameter

#### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": ""}
```

#### Response

```
400 Bad Request
{"type": "Required"}
```

### 84.4.2 Failure updating account type with invalid parameter

#### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "invalid-type"}
```

#### Response

```
400 Bad Request
{"type": "\"invalid-type\" is not one of standard, activesync, premium"}
```

### 84.4.3 Failure using inconsistent combination of allowUpgrade and type

#### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "premium", "allowUpgrade": false}
```

#### Response

```
400 Bad Request
{"message": "The allowUpgrade option is inconsistent with the provided type"}
```

### 84.4.4 Failure changing the allowUpgrade option for activesync or premium accounts

#### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"allowUpgrade": false}
```

#### Response

```
400 Bad Request
{"message": "The account is already upgraded. Please downgrade it to standard first."}
```

### 84.4.5 Failure upgrading account while allowUpgrade is set to false

#### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "premium"}
```

### Response

```
400 Bad Request
{"message": "This account is not allowed to be upgraded"}
```

## 84.4.6 Failure upgrading account when mailbox qty for OX type is exceeded

### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "activesync"}
```

### Response

```
400 Bad Request
{"computeFault": "Number of mailboxes exceeded"}
```

## 84.4.7 Success updating account type

### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "premium"}
```

### Response

```
204 No Content
```

## 84.4.8 Failure downgrading account due to current usage greater than new quota

### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "standard"}
```

### Response

```
400 Bad Request
{"message": "Current mailbox usage is larger than the new quota value"}
```

## 84.4.9 Success updating resetPassword flag

### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"resetPassword": true}
```

### Response

```
204 No Content
```

## 84.4.10 Success updating timezone, language and theme

### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"theme": "new_theme", "language": "en_US", "timezone": "America/Chicago"}
```

### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 85 Accounts-account-email-domain-username-mailboxName:GET

## 86 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}

Details about a username

### 86.1 Contents

- 1 GET /accounts/{account}/email/{domain}/usernames/{mailboxName}
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Exchange scenario
    - ◇ 1.3.2 MFG2 scenario
  - ◆ 1.4 See also

### 86.2 Request

GET /accounts/{account}/email/{domain}/usernames/{mailboxName}

#### 86.2.1 Request Parameters

account - *string*  
The user account owning the domain and the username

domain - *string*  
The domain on which the username resides

mailboxName - *string*  
Username for which to display the details

### 86.3 Response

#### 86.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name, account or username does not exist.

#### 86.3.2 Response Body

```
{
  "exchange": [
    {
      "href": "{{APIBaseURL}}/accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/",
      "rel": "related"
    }
  ],
  "mailbox": [
    {
      "href": "{{APIBaseURL}}/accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/",
      "rel": "related"
    }
  ],
  "secrets": [
    {
      "href": "{{APIBaseURL}}/accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets/",
      "rel": "related"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/{account}/email/{domain}/usernames/{mailboxName}/",
      "rel": "self"
    }
  ]
}
```

##### 86.3.2.1 Parameters

exchange - *string*  
(OPTIONAL)  
Hypermedia to the exchange details for the username. Will only show if the username is hosted exchange

mailbox - *string*  
(OPTIONAL)  
Hypermedia to the regular mailbox details for the username. Will only show if the username is a MFG2 mailbox

secrets - *string*  
Hypermedia to the secret questions for the username. Will be available for both hosted exchange and MFG2

links - *string*  
Hypermedia for this resource. Links to self

## 86.4 Examples

### 86.4.1 Exchange scenario

#### Request

```
GET /accounts/test-account123/email/test.com/usernames/exchange-mbox
```

#### Response

```
{
  "exchange": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/exchange-mbox/exchange/",
      "rel": "related"
    }
  ],
  "secrets": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/exchange-mbox/secrets/",
      "rel": "related"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/exchange-mbox/",
      "rel": "self"
    }
  ]
}
```

### 86.4.2 MFG2 scenario

#### Request

```
GET /accounts/test-account123/email/test.com/usernames/mfg-mbox
```

#### Response

```
{
  "mailbox": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/mfg-mbox/mailbox/",
      "rel": "related"
    }
  ],
  "secrets": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/mfg-mbox/secrets/",
      "rel": "related"
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/mfg-mbox/",
      "rel": "self"
    }
  ]
}
```

## 86.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 87 Accounts-account-email-domain-username:GET



## 88 GET /accounts/{account}/email/{domain}/usernames

Retrieves a list of mailboxes for a specific account and domain

### 88.1 Contents

- 1 GET /accounts/{account}/email/{domain}/usernames
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 URI Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
    - ◇ 1.3.2 Request with webmailAccess
    - ◇ 1.3.3 Paginated request with sort order
  - ◆ 1.4 See also

### 88.2 Request

GET /accounts/{account}/email/{domain}/usernames

#### 88.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

#### 88.2.2 URI Parameters

filterUsername - *string*  
Only mailboxes with username matching the value of the parameter will be returned. The values used could be specified as a partial match with a wildcard(\*).

page - *integer*  
Specifies which page should be displayed. Requires a pageSize parameter to also be provided

pageSize - *integer*  
Specifies the number of entries to be displayed on a page. Requires a page parameter to also be provided

sortField - *string*  
Specifies if the resulting list is to be sorted by a given field. Allowed values are **username** and **quota**

sortOrder - *string*  
The order in which the sorting is to be done. Allowed values are **asc** and **desc**

details - *integer*  
If set to **1** the type of mailbox users will be displayed for each mailbox

### 88.3 Response

#### 88.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name or account does not exist.

#### 88.3.2 Response Body

```
{
  "totalCount": "{count}",
  "list": [
    {
      "username": "{username}",
      "webmailAccess": "{webmailAccess}",
      "links": [
        {
          "href": "{href}",
          "rel": "{rel}"
        }
      ],
      "type": {type},
      "quota": {quota},
      "master": {master},
      "trialType": "{trialType}",
    }
  ]
}
```

```

        "trialExpiration": "{trialExpiration}",
        "allowUpgrade": {allowUpgrade}
    },
}
}

```

### 88.3.2.1 Parameters

total count - *integer*

The total number of mailboxes under the domain

username - *string*

The username of an individual mailbox

master - *integer*

Allowed values are **0** or **1**

quota - *integer*

The quota of the mailbox in Megabytes

webmailAccess - *string*

(OPTIONAL)

The type of webmail of the mailbox. Allowed values can be obtained from the [webmailNames API endpoint](#). Will only be displayed if the user passed the details URI parameter

trialType - *string*

Trial account type set for a given period of time. Supported values can be obtained from the [webmailNames API endpoint](#).

trialExpiration - *date* - YYYY-mm-dd

Expiration date of trial period formatted as YYYY-mm-dd.

href - *string*

Link to other resources relevant to the mailboxes lists

rel - *string*

Type of relation to the resource for the provided link

## 88.4 Examples

### 88.4.1 Success scenario

#### Request

```
GET accounts/test-account123/email/test.com/usernames
```

#### Response

```

{
  "totalCount": 1,
  "list": [
    {
      "username": "test",
      "links": [
        {
          "href": "{APIBaseURL}/accounts/test-account123/email/test.com/usernames/test/mailbox/",
          "rel": "self"
        }
      ],
      "type": "imap",
      "quota": 10,
      "master": false,
      "trialType": "premium",
      "trialExpiration": "2018-10-28",
      "allowUpgrade": true
    }
  ]
}

```

### 88.4.2 Request with webmailAccess

#### Request

```
GET accounts/test-account123/email/test.com/usernames?details=1
```

#### Response

```

{
  "totalCount": 1,
  "list": [
    {
      "username": "test",
      "webmailAccess": "standard",
      "links": [
        {
          "href": "{APIBaseURL}/accounts/test-account123/email/test.com/usernames/test/mailbox/",
          "rel": "self"
        }
      ],
      "type": "imap",
      "quota": 10,
      "master": false,
      "trialType": "premium",
    }
  ]
}

```

```

        "trialExpiration": "2018-10-28",
        "allowUpgrade": true
    }
}

```

### 88.4.3 Paginated request with sort order

#### Request

GET accounts/test-account123/email/test.com/usernames?page=2&pageSize=2&sortField=username&sortOrder=desc

#### Response

```

{
  "totalCount": 6,
  "list": [
    {
      "username": "test2",
      "links": [
        {
          "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/test2/mailbox/",
          "rel": "self"
        }
      ],
      "type": "imap",
      "quota": 10,
      "master": true,
      "trialType": "premium",
      "trialExpiration": "2018-10-28",
      "allowUpgrade": true
    },
    {
      "username": "test1",
      "links": [
        {
          "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames/test1/mailbox/",
          "rel": "self"
        }
      ],
      "type": "imap",
      "quota": 10,
      "master": false,
      "trialType": "premium",
      "trialExpiration": "2018-10-28",
      "allowUpgrade": true
    }
  ],
  "links": [
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames?page=1&pageSize=2",
      "rel": "first"
    },
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames?page=1&pageSize=2",
      "rel": "previous"
    },
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames?page=3&pageSize=2",
      "rel": "next"
    },
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames?page=3&pageSize=2",
      "rel": "last"
    },
    {
      "href": "{{APIBaseURL}}/accounts/test-account123/email/test.com/usernames?page=2&pageSize=2",
      "rel": "self"
    }
  ]
}

```

### 88.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 89 Accounts-account-email-domain-webmail:GET

## 90 GET /accounts/{account}/email/{domain}/webmail

Gets OpenXchange account type for the domain.

### 90.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting account type

### 90.2 Request

GET /accounts/{account}/email/{domain}/webmail

#### 90.2.1 URI Parameters

account - *string*  
domain - *string*

#### 90.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 90.3 Response

#### 90.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

#### 90.3.2 Response Body

```
{
  "type": "{type}",
  "allowUpgrade": {allowUpgrade}
  "trialType": "{trialType}",
  "trialExpiration": "{trialExpiration}",
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/webmail/",
      "rel": "self"
    }
  ]
}
```

##### 90.3.2.1 Parameters

type - *string*  
Webmail account type. Supported values can be obtained from the [webmailNames API endpoint](#).

allowUpgrade - *boolean*  
Webmail allowUpgrade option value.

trialType - *string* - Can be null  
Trial account type. Null if not set. Supported values can be obtained from the [webmailNames API endpoint](#).

trialExpiration - *date*  
Expiration date of trial period formatted as YYYY-mm-dd. Null if not set.

## 90.4 Examples

### 90.4.1 Success getting account type

#### Request

```
GET /accounts/test/email/test.com/webmail
```

#### Response

```
200 OK
```

```
{
  "type": "standard",
  "allowUpgrade": true,
  "trialType": "premium",
  "trialExpiration": "2018-10-28",
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}/webmail/",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 91 Accounts-account-email-domain-webmail:PUT

## 92 PUT /accounts/{account}/email/{domain}/webmail

Change the OpenXchange allowUpgrade option.

### 92.1 Contents

- 1 PUT /accounts/{account}/email/{domain}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating the domain settings without any options
    - ◇ 1.3.2 Success updating the allowUpgrade value

### 92.2 Request

```
PUT /accounts/{account}/email/{domain}/webmail
```

#### 92.2.1 URI Parameters

account - *string*  
domain - *string*

#### 92.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

#### 92.2.3 Request Body

```
{  
  "allowUpgrade": {allowUpgrade}  
}
```

##### 92.2.3.1 Parameters

allowUpgrade - *boolean*  
OpenXchange allowUpgrade option value.

### 92.3 Response

#### 92.3.1 Status Code

204 No Content

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

409 Conflict  
Upgrade is disabled at the product level and cannot be turned on

### 92.4 Examples

#### 92.4.1 Failure updating the domain settings without any options

##### Request

```
PUT /accounts/test/email/test.com/webmail  
{}
```

##### Response

```
400 Bad Request
```



```
{"allowUpgrade": "Required"}
```

## 92.4.2 Success updating the allowUpgrade value

### Request

```
PUT /accounts/test/email/test.com/webmail  
{  
  "allowUpgrade": true  
}
```

### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 93 Accounts-account-email-domain-webmailLimits:GET

## 94 GET /accounts/{account}/email/{domain}/webmailLimits

Getting maximum allowed number of mailboxes for each type

**Note:** /email/{domain}/webmailLimits is a shortcut equivalent of this resource

### 94.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/webmailLimits
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 See also

### 94.2 Request

GET /accounts/{account}/email/{domain}/webmailLimits

#### 94.2.1 URI Parameters

account - *string*  
Account number for the owner of the email domain entity.

domain - *string*  
Target domain name of the email domain resource.

#### 94.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 94.3 Response

Returns dictionary which contains the webmail upgrade settings for the target domain as a key-value pairs.

#### 94.3.1 Status Code

200 OK  
Success

#### 94.3.2 Response Body

```
{
  "webmailLimits": {
    "activesync": 2,
    "premium": -1,
    "starter": -1,
    "standard": -1
  },
  "links": [
    {
      "href": "https://api.hostway.com/accounts/accNum123/email/example.com/webmailLimits/",
      "rel": "self"
    }
  ]
}
```

##### 94.3.2.1 Parameters

webmailLimits - *dictionary*  
Specifies a limit for each available webmail type. Negative values and zero mean that there is no limit for the type

links - *list*  
[Hypermedia](#) for the resource.

### 94.4 See also

- [Email API](#)
- [Common Features](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 95 Accounts-account-email-domain-webmailNames:GET

## 96 GET /accounts/{account}/email/{domain}/webmailNames

Retrieves mapping of the allowed OX types with WebMail plan names for the target email domain resource. The result can be tied to a specific plan configuration on the product level.

**Note:** /email/{domain}/webmailNames is a shortcut equivalent of this resource

### 96.1 Contents

- 1 GET  
/accounts/{account}/email/{domain}/webmailNames
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 See also

### 96.2 Request

GET /accounts/{account}/email/{domain}/webmailNames

#### 96.2.1 URI Parameters

account - *string*  
Account number for the owner of the email domain entity.

domain - *string*  
Target domain name of the email domain resource.

#### 96.2.2 Request Headers

Authorization - *HTTP Authorization header [1]*  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 96.3 Response

Returns dictionary which contains the mapping of supported OpenXchange types by text key and WebMail plan names for the specific email domain resource.

#### 96.3.1 Status Code

200 OK  
Success

#### 96.3.2 Response Body

```
{
  "webmailNames": {
    "activesync": "SMX Sync Advanced",
    "premium": "SMA Collaborate Premium",
    "standard": "SMX Mail Basic"
  },
  "links": [
    {
      "href": "https://api.hostway.com/accounts/accNum123/email/example.com/webmailNames/",
      "rel": "self"
    }
  ]
}
```

##### 96.3.2.1 Parameters

webmailNames - *dictionary*  
Dictionary containing the supported OX types by text key and the WebMail plan names.

links - *list*  
[Hypermedia](#) for the resource.

### 96.4 See also

- [Email API](#)
- [Common Features](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 97 Accounts-account-email-domain-webmailUpgrade:GET

## 98 GET /accounts/{account}/email/{domain}/webmailUpgrade

Retrieves the product level upgrade configuration

**Note:** /email/{domain}/webmailUpgrade is a shortcut equivalent of this resource

### 98.1 Contents

- 1 GET /accounts/{account}/email/{domain}/webmailUpgrade
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 See also

### 98.2 Request

GET /accounts/{account}/email/{domain}/webmailUpgrade

#### 98.2.1 URI Parameters

account - *string*  
Account number for the owner of the email domain entity.  
domain - *string*  
Target domain name of the email domain resource.

#### 98.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 98.3 Response

Returns dictionary which contains the webmail upgrade settings for the target domain as a key-value pairs.

#### 98.3.1 Status Code

200 OK  
Success

#### 98.3.2 Response Body

```
{
  "allowed": false,
  "links": [
    {
      "href": "https://api.hostway.com/accounts/accNum123/email/example.com/webmailUpgrade/",
      "rel": "self"
    }
  ]
}
```

##### 98.3.2.1 Parameters

allowed - *boolean*  
Whether upgrades are enabled or disabled by the product

links - *list*  
[Hypermedia](#) for the resource.

### 98.4 See also

- [Email API](#)
- [Common Features](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 99 Accounts-account-email-domain:GET



# 100 GET /accounts/{account}/email/{domain}

Gets email domain's info.

## 100.1 Contents

- 1 GET /accounts/{account}/email/{domain}
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting domain info

## 100.2 Request

GET /accounts/{account}/email/{domain}

### 100.2.1 URI Parameters

account - *string*  
domain - *string*

### 100.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
The [Authentication](#) credentials of the client application.

Content-Type  
Optional. Set this header to `application/json; charset=UTF-8`

## 100.3 Response

### 100.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

### 100.3.2 Response Body

```
{
  "quota": 19320,
  "links": [
    {
      "location": "{APIBaseURL}/accounts/{account}/email/{domain}",
      "rel": "self"
    }
  ]
}
```

#### 100.3.2.1 Parameters

quota - *int*  
Domain quota in MB(sum of all domain's mailboxes quota).

## 100.4 Examples

### 100.4.1 Success getting domain info

#### Request

GET /accounts/test-account/email/test.com

## Response

200 OK

```
{
  "quota": 19320,
  "links": [
    {
      "href": "{APIBaseURL}/accounts/test-account/email/test.com/",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

# 101 Email API

## 101.1 Mailboxes

- GET /accounts/{account}/email/{domain}/usernames - Retrieves a list of mailboxes
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName} - Details about a username
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox - Gets mailbox details
- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox - Create or update mailbox data
- DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox - Deletes a mailbox
- POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/action - Apply action on a mailbox, e.g. reset password

## 101.2 Exchange

- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange - Create or update exchange mailbox
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange - Get exchange mailbox info
- DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange - Delete exchange mailbox
- POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/exchange/action - Apply action on exchange mailbox, e.g. reset password

## 101.3 Aliases

- GET /accounts/{account}/email/{domain}/aliases - Retrieves a list of domain aliases

## 101.4 Forwards

- POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/forwards - Sets mailbox forwarding
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/forwards - Gets mailbox forwarding info
- DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/forwards/{fwd-mailbox} - Deletes mailbox forwarding

## 101.5 Autoresponder

- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/autoresponder - Sets an auto-responder
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/autoresponder - Gets auto-responder info

## 101.6 CleanMailPlus

- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/cleanmailplus - Gets mailbox's CleanMailPlus info
- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/cleanmailplus - Updates mailbox's CleanMailPlus info

## 101.7 Webmail

- GET /accounts/{account}/email/{domain}/webmail - Getting OX allowUpgrade option for the domain
- PUT /accounts/{account}/email/{domain}/webmail - Change the OX allowUpgrade option for the domain
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail - Getting OX account
- GET /email/{domain}/usernames/{mailboxName}/webmail - Getting OX account
- POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail - Create OX account
- POST /email/{domain}/usernames/{mailboxName}/webmail - Create OX account
- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/webmail - Update OX account
- PUT /email/{domain}/usernames/{mailboxName}/webmail - Update OX account
- GET /accounts/{account}/email/{domain}/webmailLimits - Getting maximum allowed number of mailboxes for each type
- GET /accounts/{account}/email/{domain}/webmailUpgrade - Getting OX allowUpgrade option for the product
- GET /email/{domain}/webmailUpgrade - Getting OX allowUpgrade option for the product

## 101.8 Webmail Names

- GET /accounts/{account}/email/{domain}/webmailNames - Returns mapping of allowed OX types with webmail names for the domain
- GET /email/{domain}/webmailNames - Returns mapping of allowed OX types with webmail names for the domain

## 101.9 Black/White List

- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/blackwhitelist - Retrieves black/whitelisted addresses
- PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/blackwhitelist - Adds address(es) to black/white list
- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/blackwhitelist/{address} - Get particular address info
- DELETE /accounts/{account}/email/{domain}/usernames/{mailboxName}/blackwhitelist/{address} - Deletes an address from black/white list

## 101.10 InfoStore

- GET /accounts/{account}/email/{domain}/infostore - Getting InfoStore usage
- PUT /accounts/{account}/email/{domain}/infostore - Change InfoStore quota

## 101.11 Mailbox domain statistics

- GET /accounts/{account}/email/{domain}/statistics/mailboxTypeCount - Retrieves per domain statistics about webmail usage

## 101.12 Mailbox migration failure

- POST /accounts/{account}/email/{domain}/usernames/{mailboxName}/migration - Creating support ticket for failed mailbox migration

## 101.13 Mailbox Message Count

- GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/messagestats - Get mailbox message count for INBOX, SENT and SPAM folders

## 101.14 Exchange Message Count

- [GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/mailbox/exchange/messagestats](#) - Get exchange message count for INBOX, SENT and SPAM folders

## 101.15 Email Domain

- [GET /accounts/{account}/email/{domain}](#) - Gets specific email domain's info

## 101.16 Domain Black/White List

- [GET /accounts/{account}/email/{domain}/blackwhitelist](#) - Retrieves black/whitelisted addresses for domain
- [PUT /accounts/{account}/email/{domain}/blackwhitelist](#) - Adds address(es) to black/white list for domain
- [GET /accounts/{account}/email/{domain}/blackwhitelist/{address}](#) - Get particular address info
- [DELETE /accounts/{account}/email/{domain}/blackwhitelist/{address}](#) - Deletes an address from black/white list

## 101.17 Domain CleanMailPlus

- [GET /accounts/{account}/email/{domain}/cleanmailplus](#) - Gets domain's CleanMailPlus info
- [PUT /accounts/{account}/email/{domain}/cleanmailplus](#) - Updates domain's CleanMailPlus info

## 101.18 Secret Questions

- [GET /accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets](#) - Retrieves a list of mailbox(alias) secret questions
- [PUT /accounts/{account}/email/{domain}/usernames/{mailboxName}/secrets](#) - Set or update all secret questions and answers for mailbox(alias)

## 101.19 Secret Question Settings

- [GET /email/secretQuestionSettings](#) - Get franchise specific secret question settings (e.g. **minimum** required number of secret questions, etc.)

## 101.20 Supported Secret Questions

- [GET /email/secretQuestions](#) - Get dictionary of supported secret questions with text keys

## 101.21 Antivirus Antispam

- [GET /accounts/{accountId}/avas/{domain}/SSOLink](#) - Returns a SSO Link for antispam control panel

### 101.21.1 Templates

- [resetPassword](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**102 Email-domain-username-mailboxName-exchange-action:POST**

# 103 POST /email/{domain}/usernames/{mailboxName}/exchange/action

Apply an action on exchange mailbox e.g. **reset password**

## 103.1 Contents

- 1 POST  
/email/{domain}/usernames/{mailboxName}/exchange/action
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
  - ◆ 1.2 Response
    - ◇ 1.2.1 Expected Response Codes
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Request password reset of exchange mailbox
    - ◇ 1.3.2 Request password reset of exchange mailbox with wrong answer
  - ◆ 1.4 See also

## 103.2 Request

POST /email/{domain}/usernames/{mailboxName}/exchange/action

### 103.2.1 Request Parameters

account - *string*  
The user account owning the domain and the exchange mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the exchange mailbox.

### 103.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 103.2.3 Request Body

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": [{"tk": "{answer}"},
    "password": "{password}"
  ]
}
```

action - *string*  
The action that should be performed. Allowed values are:  
**resetPassword** - Reset the password of exchange mailbox.

actionData - *dictionary (Required)*  
Data to be send with the action.

secretQuestionsAnswers - *dictionary (Required)*  
Dictionary of key/value pairs, each of which represents **secret question text key/answer** pair.

tk - *string (Required)*  
The text key for the secret question. There should be no duplicate text keys per exchange mailbox.

answer - *string (Required)*  
The answer in plain-text - must be at least two characters.

password - *string (Required)*  
The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.

Password validation rules:

- Password must not be used more than once
- Minimum length 8
- Maximum length 128
- Should have at least one digit
- Should have at least one capital letter
- Should have at least one lower case letter
- Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)
- Should not start with the phrase 'pass' case-insensitively
- Should not start with 'abc' case-insensitively
- Should not start with 123
- Must not contain second-level domain, username or start with the first 3 letters from the username

## 103.3 Response

### 103.3.1 Expected Response Codes

204 No Content  
Success

400 Bad Request  
The supplied action or the supplied data is invalid.

404 Not Found  
The resource does not exist.

409 Conflict  
Validation of the secret answer failed.

## 103.4 Examples

### 103.4.1 Request password reset of exchange mailbox

#### Request

```
POST /email/test.com/usernames/mailbox123/exchange/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk1": "answer123"},
    "password": "pWord123$"
  }
}
```

#### Response

204 No Content

### 103.4.2 Request password reset of exchange mailbox with wrong answer

#### Request

```
POST /email/test.com/usernames/mailbox123/exchange/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswersList": {"tk1": "wrongAnswer"},
    "password": "pWord123$"
  }
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "message": "Invalid secret questions and answers provided",
    "code": 409,
  }
}
```

## 103.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 104 Email-domain-username-mailboxName-mailbox-action:POST



# 105 POST /email/{domain}/usernames/{mailboxName}/mailbox/action

Apply an action on a mailbox e.g. **reset password**

## 105.1 Contents

- 1 POST  
/email/{domain}/usernames/{mailboxName}/mailbox/action
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
  - ◆ 1.2 Response
    - ◇ 1.2.1 Expected Response Codes
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Request password reset of a mailbox
    - ◇ 1.3.2 Request password reset of a mailbox with wrong answer
  - ◆ 1.4 See also

## 105.2 Request

POST /email/{domain}/usernames/{mailboxName}/mailbox/action

### 105.2.1 Request Parameters

account - *string*  
The user account owning the domain and the mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the mailbox.

### 105.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 105.2.3 Request Body

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": [{"tk": "{answer}"},
    "password": "{password}"
  ]
}
```

action - *string*  
The action that should be performed. Allowed values are:  
**resetPassword** - Reset the password of a mailbox.

actionData - *dictionary (Required)*  
Data to be send with the action.

secretQuestionsAnswers - *dictionary (Required)*  
Dictionary of key/value pairs, each of which represents **secret question text key/answer** pair.

tk - *string (Required)*  
The text key for the secret question. There should be no duplicate text keys per mailbox.

answer - *string (Required)*  
The answer in plain-text - must be at least two characters.

password - *string (Required)*  
The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.

Password validation rules:

Password must not be used more than once

Minimum length 8

Maximum length 128

Should have at least one digit

Should have at least one capital letter

Should have at least one lower case letter

Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)

Should not start with the phrase 'pass' case-insensitively

Should not start with 'abc' case-insensitively

Should not start with 123

Must not contain second-level domain, username or start with the first 3 letters from the username

## 105.3 Response

### 105.3.1 Expected Response Codes

204 No Content  
Success

400 Bad Request  
The supplied action or the supplied data is invalid.

404 Not Found  
The resource does not exist.

409 Conflict  
Validation of the secret answer failed.

## 105.4 Examples

### 105.4.1 Request password reset of a mailbox

#### Request

```
POST /email/test.com/usernames/mailbox123/mailbox/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk1": "answer123"},
    "password": "pWord123$"
  }
}
```

#### Response

204 No Content

### 105.4.2 Request password reset of a mailbox with wrong answer

#### Request

```
POST /email/test.com/usernames/mailbox123/mailbox/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswersList": {"tk1": "wrongAnswer"},
    "password": "pWord123$"
  }
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "message": "Invalid secret questions and answers provided",
    "code": 409,
  }
}
```

## 105.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

# 106 Email-domain-username-mailboxName-secrets:GET

## 107 GET /email/{domain}/usernames/{mailboxName}/secrets

Retrieves a list of text keys for all mailbox(alias) secret questions.

### 107.1 Contents

- 1 GET /email/{domain}/usernames/{mailboxName}/secrets
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
  - ◆ 1.4 See also

### 107.2 Request

GET /email/{domain}/usernames/{mailboxName}/secrets

#### 107.2.1 Request Parameters

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the mailbox

#### 107.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 107.3 Response

#### 107.3.1 Status Code

200 OK  
Success

404 Not Found  
The domain name or mailbox does not exist.

#### 107.3.2 Response Body

```
{
  "list": [ "{tk}", "{tk}", "{tk}" ],
  "links": [
    {
      "href": "https://api.hostway.com/email/{domain}/usernames/{mailboxName}/secrets",
      "rel": "self"
    }
  ]
}
```

##### 107.3.2.1 Parameters

list - 'list'  
A list of secret questions of the account

tk - *string* (Unique)  
Unique secret question text key.

links - 'list'  
[Hypermedia](#) to the account secret questions resource

### 107.4 Examples

#### 107.4.1 Success scenario

##### Request

GET /email/test.com/usernames/test-mailbox/secrets

##### Response

```
{
  "list": [ "tk1", "tk2", "tk3" ],
  "links": [
    {
      "href": "https://api.hostway.com/email/test.com/usernames/test-mailbox/secrets",
      "rel": "self"
    }
  ]
}
```

}

## 107.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

**108 Email-domain-username-mailboxName-secrets:PUT**

# 109 PUT /email/{domain}/usernames/{mailboxName}/secrets

Set or update all secret questions and answers for mailbox(alias).

## 109.1 Contents

- 1 PUT /email/{domain}/usernames/{mailboxName}/secrets
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Body
      - 1.1.2.1 Parameters
    - ◇ 1.1.3 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success scenario
    - ◇ 1.3.2 Set only two secret questions and answers, when the required minimum is three
  - ◆ 1.4 See also

## 109.2 Request

PUT /email/{domain}/usernames/{mailboxName}/secrets

### 109.2.1 Request Parameters

domain - *string*  
The specific mailbox domain

mailboxName - *string*  
The name of the mailbox

### 109.2.2 Request Body

```
{  
  "tk": "answer",  
  "tk": "answer",  
  ...  
}
```

#### 109.2.2.1 Parameters

tk - *string* (Required)  
The text key for the secret question. There should be no duplicate text keys per mailbox.

answer - *string* (Required)  
The answer in plain-text - must be at least two characters.

### 109.2.3 Request Headers

Authorization - *HTTP Authorization header* [1]  
The *Authentication* credentials of the client application.

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

## 109.3 Response

### 109.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the number of the secret questions and answers pairs in the request is lower then the *required minimum*.

401 Unauthorized  
The supplied credentials are invalid.

403 Forbidden  
The authorized user does not have permissions for this operation.

404 Not Found  
The domain name, mailbox or account does not exist.

## 109.4 Examples

### 109.4.1 Success scenario

#### Request

```
PUT /email/test.com/usernames/test-mailbox/secrets  
  
{  
  "tk1": "answer1",  
  "tk2": "answer2",
```

```
    "tk3": "answer3"  
  }  
}
```

## Response

204 No Content

### 109.4.2 Set only two secret questions and answers, when the required minimum is three

#### Request

```
PUT /email/test.com/usernames/test-mailbox/secrets
```

```
{  
  "tk1": "answer1",  
  "tk2": "answer2"  
}
```

#### Response

400 Bad Request

```
"details": {"": "Shorter than minimum length 3"}
```

### 109.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)



## 110 Email-domain-username-mailboxName-webmail:GET

# 111 GET /email/{domain}/usernames/{mailboxName}/webmail

Gets OpenXchange account type.

## 111.1 Contents

- 1 GET /email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting account type

## 111.2 Request

GET /email/{domain}/usernames/{mailboxName}/webmail

### 111.2.1 URI Parameters

domain - *string*

mailboxName - *string*

### 111.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

Use the mailbox owner credentials for the specific mailbox, e.g. username: email@domain.com, password: secret

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

## 111.3 Response

### 111.3.1 Status Code

200 OK  
Success

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

### 111.3.2 Response Body

```
{
  "type": "{type}",
  "allowUpgrade": {allowUpgrade},
  "theme": "{theme}",
  "language": "{language}",
  "timezone": "{timezone}",
  "links": [
    {
      "location": "{APIBaseURL}/email/{domain}/usernames/{mailboxName}/webmail/",
      "rel": "self"
    }
  ]
}
```

#### 111.3.2.1 Parameters

allowUpgrade - *boolean*

OpenXchange allowUpgrade option value.

type - *string* - one of "standard", "activesync" or "premium"

OpenXchange account type.

theme - *string*

Theme identifier for setting a specific theme when creating an OX account

language - *string*

OpenXchange account locale of type 'en\_US'.

timezone - *string*

OpenXchange account timezone of type 'America/Chicago'.

## 111.4 Examples

### 111.4.1 Success getting account type

Request

GET /email/test.com/usernames/test/webmail

## Response

200 OK

```
{ "type": "standard", "allowUpgrade": true, "theme": "test_theme", "language": "en_US", "timezone": "America/Chicago", "links": [{"href": "{AP
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 112 Email-domain-username-mailboxName-webmail:POST

# 113 POST /email/{domain}/usernames/{mailboxName}/webmail

Creates OpenXchange account.

## 113.1 Contents

- 1 POST /email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure creating account with not-valid parameter
    - ◇ 1.3.2 Failure creating account when mailbox qty for OX type is exceeded
    - ◇ 1.3.3 Success creating account

## 113.2 Request

POST /email/{domain}/usernames/{mailboxName}/webmail

### 113.2.1 URI Parameters

domain - *string*  
mailboxName - *string*

### 113.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]  
Use the mailbox owner credentials for the specific mailbox, e.g. username: email@domain.com, password: secret

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

### 113.2.3 Request Body

```
{
  "type": "{oxtype}",
  "allowUpgrade": {allowUpgrade},
  "theme": "{theme}",
  "language": "{language}",
  "timezone": "{timezone}"
}
```

#### 113.2.3.1 Parameters

allowUpgrade - *boolean*  
OpenXchange allowUpgrade option value. The default value is true.

type - *string*  
OpenXchange account type. Supported values can be obtained from the [webmailNames API endpoint](#). If not provided - standard is default. If current mailbox qty is more than allowed for specific OpenXchange account type the response is *400 Bad Request*

theme - *string*  
Theme identifier for setting a specific theme when creating an OX account. If not provided , default theme is set.

language - *string*  
OpenXchange account locale of type 'en\_US'.

timezone - *string*  
OpenXchange account timezone of type 'America/Chicago'.

## 113.3 Response

### 113.3.1 Status Code

201 Created  
Success

400 Bad Request  
The format of the request body is invalid or the OpenXchange account type does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

409 Conflict  
OpenXchange account already exists.

404 Not Found  
The domain name or mailbox do not exist.

## 113.4 Examples

### 113.4.1 Failure creating account with not-valid parameter

#### Request

```
POST /email/test.com/usernames/john.smith/webmail
{"type": "non-valid"}
```

#### Response

```
400 Bad Request
{"type": "\"non-existing\" is not one of standard, activesync, premium"}
```

### 113.4.2 Failure creating account when mailbox qty for OX type is exceeded

#### Request

```
POST /email/test.com/usernames/john.smith/webmail
{"type": "activesync"}
```

#### Response

```
400 Bad Request
{"computeFault":"Number of mailboxes exceeded"}
```

### 113.4.3 Success creating account

#### Request

```
POST /email/test.com/usernames/john.smith/webmail
{"type": "activesync", "allowUpgrade": false, "language": "en_US", "timezone": "America/Chicago"}
```

#### Response

```
201 Created
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 114 Email-domain-username-mailboxName-webmail:PUT

# 115 PUT /email/{domain}/usernames/{mailboxName}/webmail

Upgrades/downgrades OpenXchange account type.

## 115.1 Contents

- 1 PUT /email/{domain}/usernames/{mailboxName}/webmail
  - ◆ 1.1 Request
    - ◇ 1.1.1 URI Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
      - 1.1.3.1 Parameters
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Failure updating account type with empty type
    - ◇ 1.3.2 Failure updating account type with invalid type
    - ◇ 1.3.3 Failure using inconsistent combination of allowUpgrade and type
    - ◇ 1.3.4 Failure changing the allowUpgrade option for activesync or premium accounts
    - ◇ 1.3.5 Failure upgrading account while allowUpgrade is set to false
    - ◇ 1.3.6 Failure upgrading account when mailbox qty for OX type is exceeded
    - ◇ 1.3.7 Success updating account type
    - ◇ 1.3.8 Failure downgrading account due to current usage greater than new quota
    - ◇ 1.3.9 Success updating timezone, language and theme

## 115.2 Request

PUT /email/{domain}/usernames/{mailboxName}/webmail

### 115.2.1 URI Parameters

domain - *string*  
mailboxName - *string*

### 115.2.2 Request Headers

Authorization - *HTTP Authorization header* [1]

Use the mailbox owner credentials for the specific mailbox, e.g. username: email@domain.com, password: secret

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

### 115.2.3 Request Body

```
{
  "type": "{type}",
  "allowUpgrade": {allowUpgrade},
  "theme": "{theme}",
  "language": "{language}",
  "timezone": "{timezone}"
}
```

#### 115.2.3.1 Parameters

allowUpgrade - *boolean*

OpenXchange allowUpgrade option value. If upgrades are disabled by the product, it cannot be set to true.

type - *string (Required)*

OpenXchange account type. Supported values can be obtained from the [webmailNames API endpoint](#).

The account owner (with admin-user credentials) can make changes between account types regardless of the allowUpgrade settings.

If the call is made with the end-user credentials and allowUpgrade is set to *false* providing a type different than standard results in a *400 Bad Request* response

If the call is made with the end-user credentials and type is already different than standard and allowUpgrade is set to *false* the response is *400 Bad Request*

If current usage is more than new type quota the response is *400 Bad Request*

If current mailbox qty is more than allowed for specific OpenXchange account type the response is *400 Bad Request*

If upgrades are disabled by the product, it can only be set to the current type

resetPassword - *boolean*

Forces customer to reset his password the next time he logs in to his OX account.

theme - *string*

Theme identifier for setting a specific theme when creating an OX account.

language - *string*

OpenXchange account locale of type 'en\_US'.

timezone - *string*

OpenXchange account timezone of type 'America/Chicago'.

## 115.3 Response



### 115.3.1 Status Code

204 No Content  
Success

400 Bad Request  
The format of the request body is invalid or the username does not meet the requirements.

401 Unauthorized  
The supplied credentials are invalid or do not provide permissions for this operation.

404 Not Found  
The domain name does not exist.

## 115.4 Examples

### 115.4.1 Failure updating account type with empty type

#### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": ""}
```

#### Response

```
400 Bad Request
{"type": "Required"}
```

### 115.4.2 Failure updating account type with invalid type

#### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": "invalid-type"}
```

#### Response

```
400 Bad Request
{"type": "\"invalid-type\" is not one of standard, activesync, premium"}
```

### 115.4.3 Failure using inconsistent combination of allowUpgrade and type

#### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": "premium", "allowUpgrade": false}
```

#### Response

```
400 Bad Request
{"message": "The allowUpgrade option is inconsistent with the provided type"}
```

### 115.4.4 Failure changing the allowUpgrade option for activesync or premium accounts

#### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"allowUpgrade": false}
```

#### Response

```
400 Bad Request
{"message": "The account is already upgraded. Please downgrade it to standard first."}
```

### 115.4.5 Failure upgrading account while allowUpgrade is set to false

#### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": "premium"}
```

#### Response

```
400 Bad Request
{"message": "This account is not allowed to be upgraded"}
```

## 115.4.6 Failure upgrading account when mailbox qty for OX type is exceeded

### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": "activesync"}
```

### Response

```
400 Bad Request
{"computeFault": "Number of mailboxes exceeded"}
```

## 115.4.7 Success updating account type

### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"type": "premium"}
```

### Response

```
204 No Content
```

## 115.4.8 Failure downgrading account due to current usage greater than new quota

### Request

```
PUT /accounts/test/email/test.com/usernames/john.smith/webmail
{"type": "standard"}
```

### Response

```
400 Bad Request
{"message": "Current mailbox usage is larger than the new quota value"}
```

## 115.4.9 Success updating timezone, language and theme

### Request

```
PUT /email/test.com/usernames/john.smith/webmail
{"theme": "new_theme", "language": "en_US", "timezone": "America/Chicago"}
```

### Response

```
204 No Content
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 116 Email-secretQuestions:GET

# 117 GET /email/secretQuestions

Get dictionary of supported secret questions with text keys.

## 117.1 Contents

- 1 GET /email/secretQuestions
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Headers
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting supported secret questions with text keys
    - ◇ 1.3.2 No secret questions supported

## 117.2 Request

GET /email/secretQuestions

### 117.2.1 Request Headers

Content-Type

Required. Set this header to `application/json; charset=UTF-8`

## 117.3 Response

### 117.3.1 Status Code

200 OK

Success

### 117.3.2 Response Body

Retrieves dictionary of secret questions with text keys.

```
{
  "questions":
  {
    "{tk}": "{question}",
    "{tk}": "{question}",
    ...
  },
  "links": [
  {
    "location": "https://api.hostway.com/email/secretQuestions/",
    "rel": "self"
  }
  ]
}
```

#### 117.3.2.1 Parameters

questions - *dictionary*

A dictionary of secret questions text-key with their relevant explanation in plain-text

tk - *string*

Secret question text key.

question - *string*

The question in plain-text.

links - *list*

[Hypermedia](#) to the secret questions resource

## 117.4 Examples

### 117.4.1 Success getting supported secret questions with text keys

#### Request

GET /email/secretQuestions

#### Response

200 OK

```
{
  "questions":
  {
    "tk1": "What is question one?",
    "tk2": "What is question two?",
    "tk3": "What is question three?"
  },
  "links": [
  {
    "location": "https://api.hostway.com/email/secretQuestions/",
```

```
    "rel": "self"
  }
]
}
```

## 117.4.2 No secret questions supported

### Request

GET /email/secretQuestions

### Response

200 OK

```
{
  "questions": {},
  "links": [
    {
      "location": "https://api.hostway.com/email/secretQuestions/",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 118 Email-secretQuestionSettings:GET

# 119 GET /email/secretQuestionSettings

Get franchise specific secret question settings (e.g. **minimum** required number of secret questions, etc.)

## 119.1 Contents

- 1 GET /email/secretQuestionSettings
  - ◆ 1.1 Request
  - ◆ 1.2 Response
    - ◇ 1.2.1 Status Code
    - ◇ 1.2.2 Response Body
      - 1.2.2.1 Parameters
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Success getting secret question settings

## 119.2 Request

GET /email/secretQuestionSettings

## 119.3 Response

### 119.3.1 Status Code

200 OK  
Success

### 119.3.2 Response Body

Retrieves dictionary of secret question settings.

```
{
  "minimum": 1,
  "links": [
    {
      "location": "https://api.hostway.com/email/secretQuestionSettings",
      "rel": "self"
    }
  ]
}
```

#### 119.3.2.1 Parameters

**minimum** - *integer*  
A minimum number of secret questions/answers pairs required to be on file for each email account. The minimum value for this parameter is one.

**links** - *list*  
[Hypermedia](#) to the secret question settings resource

## 119.4 Examples

### 119.4.1 Success getting secret question settings

#### Request

GET /email/secretQuestionSettings

#### Response

```
200 OK
{
  "minimum": 3,
  "links": [
    {
      "location": "https://api.hostway.com/email/secretQuestionSettings",
      "rel": "self"
    }
  ]
}
```

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)

## 120 Template:ResetPassword



## 121 POST

### {{{base}}}/email/{domain}/usernames/{mailboxName}/{{{resource}}}/action

Apply an action on{{{2}}}{1}} mailbox e.g. **reset password**

#### 121.1 Contents

- 1 POST  
{{{base}}}/email/{domain}/usernames/{mailboxName}/{{{resource}}}/action
  - ◆ 1.1 Request
    - ◇ 1.1.1 Request Parameters
    - ◇ 1.1.2 Request Headers
    - ◇ 1.1.3 Request Body
  - ◆ 1.2 Response
    - ◇ 1.2.1 Expected Response Codes
  - ◆ 1.3 Examples
    - ◇ 1.3.1 Request password reset of{{{2}}}{1}} mailbox
    - ◇ 1.3.2 Request password reset of{{{2}}}{1}} mailbox with wrong answer
  - ◆ 1.4 See also

#### 121.2 Request

POST {{{base}}}/email/{domain}/usernames/{mailboxName}/{{{resource}}}/action

##### 121.2.1 Request Parameters

account - *string*  
The user account owning the domain and the{{{1}}}} mailboxes

domain - *string*  
The specific domain for which the call will retrieve the list

mailboxName - *string*  
The name of the{{{1}}}} mailbox.

##### 121.2.2 Request Headers

Content-Type  
Required. Set this header to `application/json; charset=UTF-8`

##### 121.2.3 Request Body

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk": "{answer}"},
    "password": "{password}"
  }
}
```

action - *string*  
The action that should be performed. Allowed values are:  
**resetPassword** - Reset the password of{{{2}}}{1}} mailbox.

actionData - *dictionary (Required)*  
Data to be send with the action.

secretQuestionsAnswers - *dictionary (Required)*  
Dictionary of key/value pairs, each of which represents **secret question text key/answer** pair.

tk - *string (Required)*  
The text key for the secret question. There should be no duplicate text keys per{{{1}}}} mailbox.

answer - *string (Required)*  
The answer in plain-text - must be at least two characters.

password - *string (Required)*  
The password. The password should be at least 8 characters long, a mix of numbers, symbols and mixed-case letters. It should not begin with "pass", "123", "abc", should not contain the second-level domain name of the service, should not contain the username of the mailbox or start with its first 3 characters. There is a restriction using the same password more than once.

Password validation rules:

- Password must not be used more than once
- Minimum length 8
- Maximum length 128
- Should have at least one digit
- Should have at least one capital letter
- Should have at least one lower case letter
- Should contain at least one special character (any character other than lower or uppercase letters, digits and underscore "\_" is considered special character)
- Should not start with the phrase 'pass' case-insensitively
- Should not start with 'abc' case-insensitively
- Should not start with 123
- Must not contain second-level domain, username or start with the first 3 letters from the username

## 121.3 Response

### 121.3.1 Expected Response Codes

204 No Content  
Success

400 Bad Request  
The supplied action or the supplied data is invalid.

404 Not Found  
The resource does not exist.

409 Conflict  
Validation of the secret answer failed.

## 121.4 Examples

### 121.4.1 Request password reset of{{{2}}}{1}} mailbox

#### Request

```
POST {{{base_example}}}/email/test.com/usernames/mailbox123/{{{resource}}}/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswers": {"tk1": "answer123"},
    "password": "pWord123$"
  }
}
```

#### Response

204 No Content

### 121.4.2 Request password reset of{{{2}}}{1}} mailbox with wrong answer

#### Request

```
POST {{{base_example}}}/email/test.com/usernames/mailbox123/{{{resource}}}/action
Content-Type: application/json; Charset=UTF-8
```

```
{
  "action": "resetPassword",
  "actionData": {
    "secretQuestionsAnswersList": {"tk1": "wrongAnswer"},
    "password": "pWord123$"
  }
}
```

#### Response

409 Conflict

```
{
  "conflict": {
    "message": "Invalid secret questions and answers provided",
    "code": 409,
  }
}
```

## 121.5 See also

- [Email API](#)

Please enable JavaScript to view the [comments powered by Disqus](#). [blog comments powered by Disqus](#)