Integrating Applications with Parallels Panels via APS

Revision 1.0.12

Contents

CHAPTER 1

Overview

The document contains description and guidelines for integration of 3rd party SaaS applications with the Parallels Automation and Parallels Plesk Panels. The integration is based on Application Packaging Standard (APS) http://www.apsstandard.org/r/doc/package-format-specification-1.2/ technology, with the implementation details described below.

Parallels Plesk Panel supports a subset of APS 1.2 functionality. Details are described in Plesk APS Differences chapter.

CHAPTER 2

Integrating Application via APS

An application may be integrated with Parallels Automation (PA) in several different ways. An application represents a resource which may be combined with other resources in a pack and can be offered as single product or as a bundle of products or services. Parallels Automation uses APS technology for application delivery. It’s done in two models:

- Internally hosted applications: application files and user’s data are located inside of Customer’s space. APS package contains application files and provisioning logic. Depending on the space, there are two alternatives:
    - Shared hosting applications – being deployed in a web space.
    - VPS hosting (dedicated) applications – being deployed in a Parallels Virtuozzo Containers http://www.parallels.com/products/pvc/ VPS.
- Externally hosted applications – application files and user’s data are located outside of Customer’s space, at dedicated multi-tenant application server/cluster. APS package contains provisioning logic only. The multi-tenant server may be located either in Internet or inside of PA infrastructure, but is not managed by PA.

When selecting the means of application delivery in APS, it’s necessary to consider the overall sales model and the necessary supporting technologies. For example, Java-based applications cannot be deployed in a web space and require a dedicated container. In this case application should be packaged as internally hosted VPS one or, if the application supports multi-tenancy, as externally hosted one. APS Questionnaire http://survey.apsstandard.org/index.php?sid=2 may be used to decide about right packaging method.

CHAPTER 3

Application Components

From APS point of view, an application consists of components (services): a main component which is purchased by users with application and those additional ones which are available for upselling. The main component is responsible for core functionality of the application, while additional components may refer to extra features.

Let's consider a provider who wants to use Parallels Automation with APS to resell a "Cloud Mail" service, which supports multi-tenancy (such as Hosted Exchange or Zimbra), to own users. Let's also assume the service is hosted at ISV's side and provider should pay him for all subscriptions being sold.

From provider's point of view, the APS application should enable selling of mail service with predefined number of mailboxes of 5GB and 25GB. The service should cost zero, but its mailboxes should cost $10 and $15 per month correspondently. User should be able to purchase additional mailboxes at any time. Additionally user should be able to see consumption of disk space for each particular mailbox and for all mailboxes as well. As opposed to provider, the ISV of the Cloud Mail service does not charge for mailboxes, but charges for mail service subscription and total disk space of its mailboxes: $20 per month for service and $2 per GB per month for disk space. It is explained below how these objects can be described in APS package.

First of all, APS package cannot contain any billing data, such as prices, periods or payment methods for a service. This data is configured in Parallels Automation, in service plans with the application by provider and/or reseller. Prices are assigned on components which are exposed by APS package: services, resources and license keys. For more details how to assign pricing, see View of Provider > Application Configuration ("Application Configuration" on page 13).

Services

Each application component is represented by a “service” APS metadata object. APS services represent a hierarchy which reflects logical structure of application. For example, for a multi-tenant application, the top-level application service represents a tenant and a user of the application, who is created inside of the tenant, is declared as second-level service. Additionally applications may declare a number of user's features (third-level services), which may be available to a user of application.

Services from different branches of hierarchy can be refer to each other. For example, an instance of "mail list member" service can refer to particular "mailbox" service instance, which is included in the mail list.

The Cloud Mail application from the example above has to enable selling of mail service, mailboxes and anti-spam/anti-virus protection. Thus, it defines the following hierarchy of components:

- Mail organization at top-level
  - User’s mailbox at 2nd-level
    - Anti-spam/anti-virus protection for mailbox at 3rd-level.

For more details see APS Format 1.2: Application Packaging Guide http://www.apsstandard.org/r/doc/aps-format-1.2-packaging-guide/index.htm.

Resources

APS allows an application to expose its resources (or counters) to PA. For example, application may report disk space usage, number of users, number of processed requests, number of filtered emails or any other internal resource. As direct result, customers may be charged for usage of these counters and may be restricted in using of them with subscription limits. When a customer purchase additional amount of a resource, PA increase its limit and can notify application about it. Counters can be measured in **KB/MB/GB**, **Units** and **MHz**.

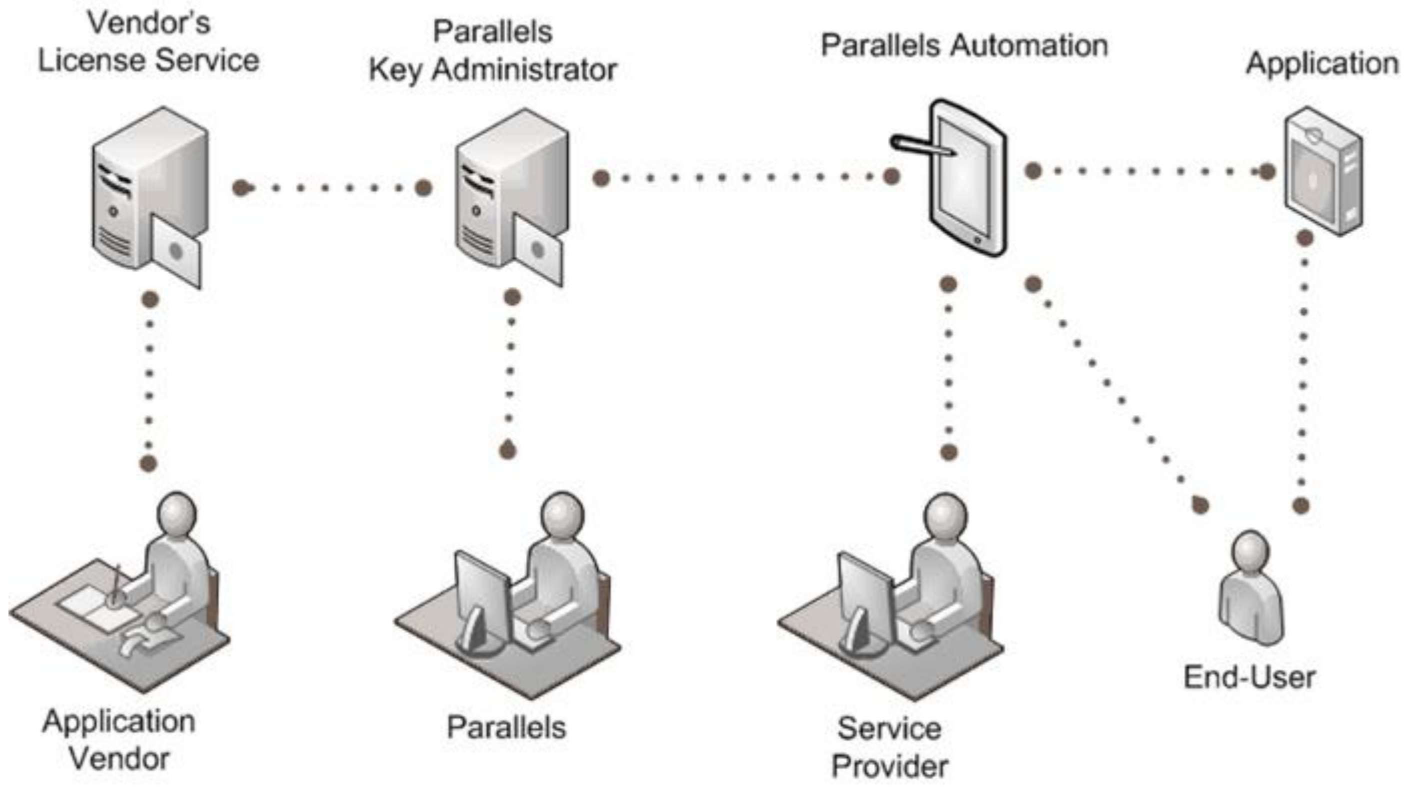
The Cloud Mail application from the example above needs to account disk space per each mailbox, thus it defines an APS resource under APS mailbox service. Aggregation capability for total mailbox disk space of a subscription is provided by PA automatically.

Additionally application counters can be measured in **MB-hours**, **Unit-hours** and **MHz-hours**. These resource classes (***-hours**) are additive. It means that usage of such resources is accounted for subscription period only, it increases during the period and it is reset at end of each period. Note, that regardless of the reset date, application should report usage of the resource from the date of application installation, POA recalculates resource usage for current period automatically. For additive resources, there are custom classes, see Appendix A ("PA-specific APS Classes" on page 25) for more details.

For more information see Resources http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.metadata.service.resources in the APS Format Specification 1.2.

License Keys

APS defines a mechanism for provisioning of license key for applications, starting from creation of license key at side of Vendor (ISV) and ending with installation of license in particular application instance. The workflow of a license key is shown in the picture below.



Upon installation of the application a license request is submitted to the Key Administrator - the component which manages all licensing-related operations. Once the license key is delivered to the application, the application becomes available to end users. License keys with different application features may be offered to customers in PA store. To upgrade an Application to a more advanced version, you have to upgrade the key.

The Cloud Mail application from the example above does not need delivery of license key, because actual provisioning of the service is done at side of ISV and, as a result, he controls number of application subscriptions.

For more details see APS Format 1.2: Licensing Aspect specification http://www.apsstandard.org/r/doc/aspect-licensing-1.2/.

CHAPTER 4

View of Customer

From high level perspective of an end-user (a Customer), Parallels Automation allows to purchase applications from Online Store and to manage them via Customer's Control Panel.

Applications in Online Store

Parallels Automation contains an online web store with the provider’s product offering. The products includes APS-enabled services (applications) and services provisioned without APS as well. The store can be configured in way that an APS application is presented as a single product or as a part of a bundle. Application product may include different number of application components and allowed features (APS services and/or licenses) and consumption limits for internal application resources (APS resources). All these components and limits can be purchased as a whole or as different parts, and by different prices as well.

Considering the example with the Cloud Mail application discussed above, the Online Store can allow a customer to purchase subscription with 10 mailboxes initially. Types of mailboxes can be selected during purchase, for example, 4 mailboxes with 5GB disk space and 6 with 25GB. At a later time (as needs grow), the customer can buy additional mailboxes of any type. Subscription is sold on one months and can be renewed later.

The Online Store handles all purchase-related activities such as displaying of application terms and conditions, charge credit cards, etc. For more details see **APS Certification Guide**.

Application Control Panel

PA provides a unified interface for all types of applications whether external or shared. This means that such applications are displayed and managed by customers in the same way. Once an application is paid and installed, a customer can add and remove application users, other services or switch to a different edition if such have been purchased and made available to the customer. All used and unused resources and services are displayed in the Control Panel.

Application resources and services defined in an APS package are represented in PA as resources of various types. A resource in terms of PA is the minimum sellable unit. By selling a resource you may sell an application as a whole or as a limited combination of services offered as an edition. This said, a resource in terms of APS are different from a resource in PA in a way that the former one defines a particular component of the application, while a Resource Type may be parameterized with usage limits, rates, etc. This way PA makes it unnecessary to define pricing, resource limits or any billing specifics in the APS package. Once a Package is successfully imported in PA, it automatically becomes a sales item and can be provided to customers based on the terms defined in the service plan.

Based on the number of licenses sold by the provider the fees become payable to the software vendor. PA supports two methods of license delivery and reporting:

- **Manual**: Licenses are issued by ISV and supplied to the Provider in bulk. The delivery method is agreed between the vendor and a provider and is beyond the scope of PA. This method is the fastest way to sell cloud services drawback of this option is that it works only with external applications.
- **Automated**: Licenses are issued through Parallels Key Administrator, which is integrated with PA. This option requires integration with the Key Administrator on the vendor end.

Ordering of Service Instances

An application may allow to order its service instances. This feature may be used for a number of different purposes, for example, a telephony service where incoming calls are distributed between subscribers. PA allows to order service instances by order numbers which are stored in a designated setting. When a value of this setting is increased or decreased by one, the respective service is placed up before the preceeding or down below the following service instance in the list.

To enable sorting of instances for a service, `order` class is used (see PA-specific APS Classes). For more details on packaging and testing see **Ordering Application Service Instances** in **APS Certification Guide**.

To enable service order, a setting of `order` class is used (see PA-specific APS Classes). This setting should also have `generate="sequence"` attribute, which means that this setting is used to store order numbers for list items.

```
<setting type="string" id="setting" class="order" generate="sequence">
	<name>Order</name>
</setting>
```

CHAPTER 5

View of Provider

Deployment of an APS application in PA by a service provider consists of two parts: preparation of hosting environment for application instances and definition of application feature sets for sale. The first part is mostly related to satisfying the technical requirements of the application. The second part is about selection of models, in which the provider wants to offer application and its components/resources to customers. The second scenario will be explored in more detail.

Application Configuration

Using PA, the provider can define applications, application components, their features and prices which will be available to customers.

Application components are configured by setting up several parameters in the application and mapped on Resource Types:

- APS global settings – parameters which are common for all application instances,
- Hidden settings of APS services – parameters of application components which may be purchased by customer as an option.

After definition of component parameters, provider may combine application components (resource types) in multiple offers (service plans). Each service plan as well as each included resource type get prices and periods on which customers can subscribe on the plan.

Considering example with the Cloud Mail application discussed above, provider configures 2 resource types for mailbox component: "Mailbox - 5GB" and "Mailbox - 25GB". The type of mailbox is defined by hidden setting of mailbox APS service. Any new/existing mailbox can be switched between the resource types which will result in change of the setting. Another resource type is created for mailbox disk space APS resource. Usage of all mailboxes will be aggregated in the resource type, but each particular mailbox will get its usage too. The last resource type is created for Cloud Mail application itself (for mail organization service).

The provider adds these resource types in a new service plan and configures one month subscription period for it. The plan itself gets zero price, because the service without mailboxes is free. Mailbox resource types get $10 and $15 prices respectively, but zero included amount and unlimited maximal one (for upselling). The resource type for a disk space is also added to the service plan without any limits.

For mutual settlement of accounts with the ISV of Cloud Mail service, the provider can see statistics with total amount of the Cloud Mail application resource type and aggregated sum of the mailbox disk space resource type among all existing subscriptions of the service plan.

Application Management

The provider can manage by availability of application upgrade for customers. Upgrade of application itself cannot be sold – if it’s enabled it’s available for all customers, but the upgrade of license keys can. Provider can change multiple application instances at once. It’s done by update of a parameter (APS global setting) in application configuration, which, in turn, reconfigures all application instances bound to it.

CHAPTER 6

View of Reseller

The reseller in PA has an account, like the provider's one, but without management permissions for hosting resources. Reseller has ability to combine applications and their components/resources, which are pre-defined by provider, into different bundles and to assign different prices to them. Reseller cannot redefine application settings or feature sets.

Parallels Automation allows Reseller to hide application identity through its own brand. It's done by customization of a URL and branding parameters of externally hosted applications. Declaration of branding support may be included in APS metadata of application. To be branded, an APS global setting must belong to setting group of class “branding”. To get a URL branded, the application must declare global setting for URL, having class “access_point”, within the group.

CHAPTER 7

Data Embedding in Application

Parallels Automation may embed some objects, which are managed from customer’s control panel, into APS applications. List of such objects includes:

- Users of customer’s account (service users)
- Mailboxes of users (Linux mail hosting)
- Domains of customer
- Details of customer’s account
- Trial flag of customer's subscription

Service Users

Customers in PA may have additional users with personal control panels and personal applications which they are allowed to use. These users can be created in the application of customers automatically, if it declares correspondent component (APS service with id=”account”). PA performs it by filling APS service settings, which have standard APS classes, see Settings Semantics http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.metadata.service.settings.semantics. Besides standard classes, PA defines custom ones and propagates additional information about a user with them. See the appendix for more details. All users’ information is synchronized between control panel and application automatically.

Mailboxes

When a user with mailbox is created in application, optional application’s components may get access to the mailbox as well. For example, voicemail service of PBX application may get access to user’s mailbox to store voice messages there. To achieve it, APS service of an ‘account’ class should have sub-service with mailbox requirement. When being provisioned, the sub-service connects to the mailbox and get the necessary data automatically, see Mail Aspect http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.aspects.mail for more information.

Domains

Applications can use customer's domains which are hosted on PA and modify DNS records on them. For example, being installed on a domain, an anti-spam application may replace MX record on the domain to the one, pointing to an external filtration server, and deliver old MX record in application for it to return clear mail traffic back to mail server. See APS Format 1.2: Application Packaging Guide http://www.apsstandard.org/r/doc/aps-format-1.2-packaging-guide/index.htm for details how to package this integration.

Externally hosted applications get domains in setting with a special class="domain-name", see Service Settings, Data Types http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.metadata.service.settings.types for details. Internally hosted applications (shared and VPS hosting) can also get name of primary domain via an environment variable of URL Mapping http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.metadata.provision.mapping.

Application domains may be used when requesting emails from Customer or Service User. If a setting has class="email-on-application-domain", then POA displays input of two parts - name and domain selection. Domains listed in selection are those that are integrated with application using setting with class="domain-name".

Environment

Parallels Automation may propagate information about a customer's environment, such as login, password or language, into the application. It's done through standard classes of settings. See Service Settings, Settings Semantics http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.metadata.service.settings.semantics for details. In addition, PA can deliver information about customer's time zone into application. It's done by setting in APS metadata with special class "tz".

Trial Subscription

PA can deliver services and applications under a trial subscription. For example, you may let users evaluate your application for one month. To enable a trial subscription for your application, you will have to add necessary settings to the application metadata (see PA-specific APS Classes) and edit the configuration script to process the trial option (application setting).

A trial subscription is provided in PBA (Parallels Business Automation). PBA propagates all changes of the subscription trial flag to the trial attribute of the POA subscription. When PBA updates the attribute, POA updates for the subscription all instances of the APS application, which has the trial setting. More specifically, POA reconfigures the application with a new value of the setting - executes the configuration script of root service after changing the subscription period.

Note: The related option (**Notify about subscription trial state changes**) must be set to "**Yes**" in PBA to synchronize the statuses of trial subscriptions between PBA and POA (see **Configuring Parallels Operations Automation** in **PBA Provider's Guide**).

Action Links

Some applications may offer features that are activated by respective entry points (action links). Such links are used to perform specific actions at an external application service. Action links may have various statuses representing the current status or operation. Embedding action links allows for easy access to application components and features from within end-user interface. For example, action links may be used in mobile management software which enables an administrator to wipe data off a phone lost by an employee. Also, a virtual machine management application may enable an administrator to remotely run and stop virtual machines. Such application will offer a list of instances, each associated with a virtual machine. Each instance will display action links used to start, stop and restart a virtual machine and also a field showing the current status. This example will serve here to show how to package an application with action links.

To add an action link to your application, you have to define two things:

- Entry points with action class.
- Each entry point has to include the following details:
  - an entry point with `action` class - notifies the Controller that this entry point is an action link.
  - a variable with `status` class - notifies the Controller that this value should substitute 'configure' in the configuration script.
  - a variable with `status_condition` class - notifies the Controller on the state when this entry point should be displayed.

Guided by these requirements, you can create an action link that, for example, start a virtual machine, as shown below. The example below is given just for reference. See inline comments for better understanding.

```
<entry-points>
       <entry class="action" dst="">
    <!-- Action class signifies that this entry point is an action link. -->
              <label>Start</label>
          <!-- Action link text displayed in the customer UI. -->

              <icon path="images/start.gif" />
          <!-- Icon displayed to the left of the link text. -->

              <variable name="status" class="action">Start</variable>
          <!-- A value passed to the configuration script instead of 'configure'. -->

              <variable name="var1" class="status">Starting</variable>
           <!-- Text replacing the status with a transient status when the action is in
progress. -->

              <variable name="var2" class="status_condition">PowerOff</variable>
          <!-- The value of the status-class variable when this action link is displayed.
-->
       </entry>
</entry-points>
```

To have an external service perform a specific action at a click of a link, a configuration script has to be executed with a specific value. This value is retrieved from a variable of `status` class defined for each action link (see above). The variable value is not passed to the script directly, rather it is submitted as a value of a setting with the `status` class (see below).

```
<setting type="enum" id="service_status" class="status"
                                 protected="true" default-value="Stopped">
                                 <name>Current Service Status</name>
                                 <name xml:lang="en-US">Current service status</name>
                                 <choice id="Running">
                                        <name>Running{color:green}</name>
                                        <name
xml:lang="en-US">Running{color:green}</name>
                                 </choice>
                                 <choice id="Failed">
                                        <name>Failed to start! Try
again.{color:red}</name>
                                        <name xml:lang="en-US">Failed to start! Try
again.{color:red}</name>
                                 </choice>
                                 <choice id="Stopped">
                                        <name>Stopped{color:red}</name>
                                        <name
xml:lang="en-US">Stopped{color:red}</name>
                                 </choice>
                                 <choice id="Starting">
                                        <name>Starting{color:yellow}</name>
                                        <name
xml:lang="en-US">Starting{color:yellow}</name>
                                 </choice>
</setting>
```

Besides starting a virtual machine you may want to enable an administrator with an ability to stop, suspend and restart VM's. In all these cases the same setting is used to take on values of various action links and thus perform various operations with VM's.

Note: When an external service has completed an action or changed a state, it may return the updated status as structured output. Parsing output values is performed by the APS Controller.

C H A P T E R 8

Plesk APS Differences

Application types

Parallels Plesk's APS integration is intended to manage shared hosting and external services only. It does not support VPS hosting applications which are deployed on a dedicated virtual server.

Structured Output

Returned results from configure scripts, referred to as structured-output, are not supported by Parallels Plesk. As an alternative, any data resulting from installation might be written to a local file or database.

DNS Aspect

Starting from Parallels Plesk 11, APS DNS aspect is supported. It allows creation or substitution of DNS records on Plesk. However, Parallels Plesk does not support the ability to set those DNS records based on values returned from the provisioning scripts as Parallels Automation does.

Direct Calls of Plesk API

Parallels Plesk offers an API for controlling a hosted account. The details can be found here http://www.parallels.com/ptn/documentation/ppp/. This API can be used to modify hosting settings, like DNS records, during APS script execution to include functionality not supported by the Plesk APS controller. The customer's account credentials for API communication can be passed to the APS package's provisioning scripts by including the following settings in the APP-META.xml file.

```
<group class="authn">
   <setting id="admin_login" class="login" type="string">
        <name>Administrator's Login</name>
   </setting>
   <setting id="admin_password" class="password" type="password"">
         <name>Password</name>
    </setting>
</group>
```

In the APS package's provisioning scripts, you can grab those values and use to call the local Plesk API, for example in PHP script:

```
$plesk_admin_username = getenv("SETTINGS_admin_login");
$plesk_admin_password = getenv("SETTINGS_admin_password");
```

Licensing Aspect

For applications requiring a license in Parallels Plesk Panel (PPP), license requirements defined in APP-META.xml should include type attribute in the following format: “urn:pla:<product-alias>:<license-version>”.

1 <product-alias> must be an alphanumeric string allowing dashes. Example: “parallels-plesk-panel”.

2 <license-version> must be a three-digit version in a dotted format, where second and third digits are optional. Examples: “9.12.6” or “9.12” or “9”.

Here is an example where this data needs to go in the full package metadata:

```
<application ...>
       ...
       <service ...>
              ...
              <requirements ...>
                     ...
                     <l:license type="urn:pla:pinnacle-cart:1" ...>
                     ...
                     </license>
                     ...
              </requirements>
              ...
       </service>
       ...
</application>
```

CHAPTER 9

Next Steps

- Read APS Format 1.2: Application Packaging Guide http://www.apsstandard.org/r/doc/aps-format-1.2-packaging-guide/index.htm, look at examples and package your application in APS
- Consult with Application Packaging Standard (APS) Format Specification v1.2 http://www.apsstandard.org/r/doc/package-format-specification-1.2/ about specific details
- Test your application with APS-compliant control panels as described in APS Package Certification Guide http://www.apsstandard.org/r/doc/APS_Package_Certification_Guide/index.htm
- Publish your application in APS Catalog (http://www.apsstandard.org/app) (contact APS team to get access)
- Ask any questions to APS Support http://www.apsstandard.org/feedback

APPENDIX A

PA-specific APS Classes

Parallels Automation supports all standard classes from Application Packaging Standard (APS) Format Specification v1.2. The list below contains non-standard classes of settings and their groups which are specific for APS Controller implementation in Parallels Automation. Being directed by these classes, PA fills correspondent settings by appropriate data automatically.

Class	APS Metadata Object	Description	Example
action	Entry point	Entry point with this class perform a specific action or change a status of an external service.	
multi-step-login	Entry point	Such entry points allow login with a token when "Embed application user interface" option is enabled. Clicking on entry point (opening a subscription with application) does two actions: 1) Verification script is called and returns entry point url with a token into a setting using 'structured-output'. The setting should be referred from entry point via"variable" element. 2) The entry point is opened as defined in APS.	<entry class="multi-step-login" dst="{cp_url}{token}" method="GET"> <label>Login with Token</label> <variable name="cp_url" value-of-setting="cp_url" /> <variable name="token" value-of-setting="token" /> </entry>

contact	Service	For Hosted Exchange and Qmail, an instance of the service with class="contact" is created as follows: - if there is a requirement for mailbox, the service is created only for mailboxes - if there is no requirement for mailbox, the service is created only for maillists - PA automatically creates/removes/updates instances of the service in case of contact changes See APS Mail Aspect http://www.apsstandard.org/r/doc/package-format-specification-1.2/index.html#s.aspects.mail	When a Customer has a subscription with Hosted Exchange or QMail resources and an APS application which defines a service with class="contact", then an instance of the service will be automatically created for every new contact created by QMail/Exchange for this APS application if the Customer selects the "Show in address book" option in his CCP, for example, when creating a mailbox.
login	Setting	Login of a User	
password	Setting	Password of a User	
locale	Setting	Locale of a User	
locality	Setting	City of a User	
country-name	Setting	Country of a User	
region	Setting	State/Province of a User	
street-address	Setting	Address line1 of a User	
extended address	Setting	Address line 2 of a User	
postal-code	Setting	Zip/Postal code of a User	
title	Setting	Title of a User	
display-name	Setting	Display name of a User	
given-name	Setting	First part of display name of a User	
family-name	Setting	Second part of display name of a User	
work	Setting	Work phone number of a User	
fax	Setting	Fax number of a User	
cell	Setting	Mobile phone number of a User	
home	Setting	Home phone number of a User	
organization-unit	Setting	Department where a User works in customer’s organization	
organization-name	Setting	Name of customer’s organization	

url	Setting	Web page of customer’s organization	
email	Setting	E-mail address of a User	
domain-name	Setting	Domain of application	
access_point	Setting	URL for entry points building. The setting value may be overridden by branded URL of application	
status	Setting	A setting used to pass variable values of action links to a configuration script.	
status_condition	Setting		
subscription_trial	Setting	Trial subscription flag (true or false).	
subscription_id	Setting	Unique subscription ID.	
tz	Setting	Time zone of customer in format of Olson database. http://en.wikipedia.org/wiki/Tz_database	
email-on-application-domain	Setting	Email account. The username part is defined by the user, and the domain part is displayed as a dropdown box and initialized with domains associated with the application.	
login-on-application-domain	Setting	Makes POA to check that service user login is on one of domains associated with the application.	
managed-by-account	Setting	Service access level that enables a Service User to activate a sub-service under a root account service (it also allows editing service settings).	
subscription-user-limit	Setting	'Application User' sub-resource limit from Customer subscription ('Application User' is a sub-resource of the 'Application' resource type)	
order	Setting	The order in which a service is distributed among users.	
branding	Setting Group	Group of settings, which values may be customized in application brand.	
instance-list	Setting Group	Group of settings, which values are displayed in list of service instances	
mb-h	Resource	Additive resource in MB-hours.	

item-h	Resource	Additive resource in Unit-hours.	
mhzh	Resource	Additive resource in MHz-hours.	

table	Setting Group	Group of settings which is displayed as a table.
		Each column represents a single setting of the “list” type, where list elements are located in rows.
		The setting names are displayed in the column titles.
		All the settings in groups are of the list type only.
		Default values for list settings can be defined as well.
		Note: All list settings must have the same values of “min-items” = “max-items” attributes.
			The following setting group are displayed as a table with 3 columns “Description”, “Action”, “Target” and 5 rows.
			
			`<group class="table">`
			`<setting id="desc" type="list" value-type="string" min-items="5" max-items="5">`
			`<name>Description</name>`
			`</setting>`
			`<setting id="action" type="list" value-type="enum" min-items="5" max-items="5">`
			`<name>Action</name>`
			`<choice id="fwd">`
			`<name>Forward</name>`
			`</choice>`
			`<choice id="vmail">`
			`<name>VoiceMail</name>`
			`</choice>`
			`</setting>`
			`<setting id="target" type="list" value-type="string" min-items="5" max-items="5">`
			`<name>Target</name>`
			`</setting>`
			`</group>`

Notes 1) User in the table may be either a service user or an administrative user.
2) If the contact information (address, city, state, country) of a service user is empty, it will be taken from Customer's account data. When the service user defines own contact information, the Customer's one must be replaced. Phone numbers must be taken only from service user's data and must be empty if not defined for the service user.

As defined in the APS Specification, settings may be `hidden` and `protected` (see **5.2.5 Service Settings** in APS Specification) These attributes are common for any APS-compliant hosting platforms, though combination of their values for the same setting may affect its display and initialization, which is specific only for PA. See example in Packaging SpamExperts Incoming Email Security Firewall Application.

Visibility	**Protected**	**Setting Value**
visibility='hidden'	unspecified	Default
visibility='hidden'	protected='true'	Initialized by the configuration script
unspecified	unspecified	Default or input by the Customer
unspecified	protected='true'	Default

Index