Prodly AppOps Integration

Table of Contents

Overview 2

Salesforce Package Application Programming Interface 2

- Deployment Web Service 2
  - Usage 2
  - Namespace 3
  - Ways to Reference the REST Endpoint 3
  - Sample URLs 3
  - Sample Command LIne Interface (CLI) Code 3
  - Related Salesforce Documentation 4
- Deployment Apex Service 4
  - Usage 4
  - Namespace 4
  - Methods 4
  - Parameters 5
  - Return Value 5
  - Sample Code 5
- API Error Messages 6

Deployment Scheduling 6

Salesforce DX Plugin 8

- Setup 8
- Usage 9
- Response 11
- Examples 11

Overview

These topics cover topics of interest to systems integrators, release engineers, and developers integrating Prodly AppOps as part of their development process. For example, for:

- Integration with a release management system
- Automatic periodic updates to refresh an existing sandbox

Prodly supports API requests for Apex and REST to accomplish these goals. Prodly also offers support for Salesforce DX integration.

Salesforce Package Application Programming Interface

This section describes the Prodly AppOps application programming interface (API). The API helps integrate continuous delivery and release management systems to automatically perform AppOps deployments. For example, use API requests to deploy a data set or deployment plan.

The AppOps API is a RESTful API that allows you to integrate AppOps deployments into a larger continuous integration process. AppOps API requests use the Force.com REST API to invoke AppOps actions. AppOps also supports global Apex services for internal Salesforce automation.

The following API requests are available:

- Deployment Web Service
- Deployment Apex Service

Deployment Web Service

Deployment Web Service is REST web service specific to Prodly AppOps that provides the ability for AppOps consumers and integrated systems outside of Salesforce to invoke deployments and perform other actions through REST–based API requests exposed by the AppOps Salesforce package in your AppOps control org.

Usage

The Deployment Web Service allows you to deploy deployment plans and data sets from applications external to the Salesforce, such as a continuous delivery or release management system. For example:

- You are running CircleCI and after deploying updated code to a QA sandbox, you need to automatically deploy the updated data to the sandbox for end to end testing.
- You have a custom built solution coordinating CPQ deployments that performs automated steps to promote product catalog changes between development, QA, and UAT sandboxes.

Namespace

PDRI

Ways to Reference the REST Endpoint

- For deployment plans:
    - /dataset/deploy/plan/*
    - https://*<org_instance>*.salesforce.com/services/apexrest/PDRI/dataset/deploy/plan/*<deployment_plan_id>*?targetConnectionId=*<connection_id>*
    - https://*<custom_domain>*.my.salesforce.com/services/apexrest/PDRI/dataset/deploy/plan/*<deployment_plan_id>*?targetConnectionId=*<connection_id>*
- For data sets:
    - /dataset/deploy/*
    - https://*<org_instance>*.salesforce.com/services/apexrest/PDRI/dataset/deploy/*<data_set_id>*?targetConnectionId=*<connection_id>*
    - https://*<custom_domain>*.my.salesforce.com/services/apexrest/PDRI/dataset/deploy/*<data_set_id>*?targetConnectionId=*<connection_id>*

Sample URLs

- https://na8.salesforce.com/services/apexrest/PDRI/dataset/deploy/a000H00400s6WikQAE?targetConnectionId=a030H00120XIWDzQAP
- https://prodlytest-dev-ed.my.salesforce.com/services/apexrest/PDRI/dataset/deploy/a000H00400s6WikQAE?targetConnectionId=a030H00120XIWDzQAP

Sample Command LIne Interface (CLI) Code

Code Format

```
$ curl -H "Authorization: Bearer <session_id>" "<endpoint>"
```

Code Sample

```
$ curl -H "Authorization: Bearer
BBASDQMFfAfpRjOw4dE5c1p0vzbeV98wgl460GrL3eOLfghnSpfFeoeqb.HH0XurSpXjC
eHAjZZJioJ1magcmixMxpED6xe0n"
"https://prodlytest-dev-ed.my.salesforce.com/services/apexrest/PDRI/d
ataset/deploy/a000H00400s6WikQAE?targetConnectionId=a030H00120XIWDzQA
P"
```

JSON Response

```
{
  "resultIds":null,"resultId":"a040H0000168iXcQAI","error":null
```

```
}
```

Related Salesforce Documentation

https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_rest.htm

Deployment Apex Service

The DeploymentService class is an Apex service specific to Prodly AppOps.

Usage

The DeploymentService class allows you to deploy deployment plans and data sets using the Apex API. Depending on the parameters you specify, the API request migrates data from your control org or a source org you specify to one or more destination orgs.

Namespace

PDRI

Methods

The DeploymentPlanService class provides the following methods for deployment plans:

- **`ID deploy(deploymentPlanId, targetConnectionId)`** – Deploys the given deployment plan, migrating data from the control org to the given destination org.
- **`ID deploy(deploymentPlanId, sourceConnectionId, targetConnectionId)`** – Deploys the given deployment plan, migrating data from the given source org to the given destination org.
- **`List<ID> deploy(deploymentPlanId, targetConnectionIds)`** – Deploys the given deployment plan, migrating data from the control org to the given list of destination orgs.
- **`List<ID> deploy(deploymentPlanId, sourceConnectionId, targetConnectionIds)`** – Deploys the given deployment plan, migrating data from the given source org to the given list of destination orgs.

The DeploymentService class provides the following methods for data sets:

- **`ID deploy(rootDataSetId, targetConnectionId)`** – Deploys the given data set, migrating data from the control org to the given destination org.
- **`ID deploy(rootDataSetId, sourceConnectionId, targetConnectionId)`** – Deploys the given data set, migrating data from the given source org to the given destination org.
- **`List<ID> deploy(rootDataSetId, targetConnectionIds)`** – Deploys the given data set, migrating data from the control org to the given list of destination orgs.

- **`List<ID> deploy(rootDataSetId, sourceConnectionId, targetConnectionIds)`** – Deploys the given data set, migrating data from the given source org to the given list of destination orgs.

Parameters

The `deployPlan()` method accepts the following parameters:

- **`ID deploymentPlanId`** – The ID of the deployment plan to deploy.
- **`ID sourceConnectionId`** – The ID of the AppOps connection to the source org from which to migrate data.
- **`ID targetConnectionId`** – The ID of the AppOps connection to the destination org that receives the migrated data.
- **`List<ID> targetConnectionIds`** – A list of IDs of the AppOps connections to up to five destination orgs that receive the migrated data.

The `deploy()` method accepts the following parameters:

- **`ID rootDataSetId`** – The ID of the data set to deploy.
- **`ID sourceConnectionId`** – The ID of the AppOps connection to the source org from which to migrate data.
- **`ID targetConnectionId`** – The ID of the AppOps connection to the destination org that receives the migrated data.
- **`List<ID> targetConnectionIds`** – A list of IDs of the AppOps connections to up to five destination orgs that receive the migrated data.

Return Value

Depending on the parameters you pass in, `deployPlan()` and `deploy()` return one of the following values:

- **`ID`** – The ID of the result record.
- **`List<ID>`** – A list of result record IDs, one for each destination org.

Sample Code

```
ID rootDataSetId = [SELECT Id from PDRI__DataSet__c where Name =
'<data_set_name>' LIMIT 1].Id;
ID targetConnectionId = [SELECT Id from PDRI__Connection__c where
Name = '<connection_name>' LIMIT 1].Id;
ID deploymentResultId = PDRI.DeploymentService.deploy(rootDataSetId,
targetConnectionId);
```

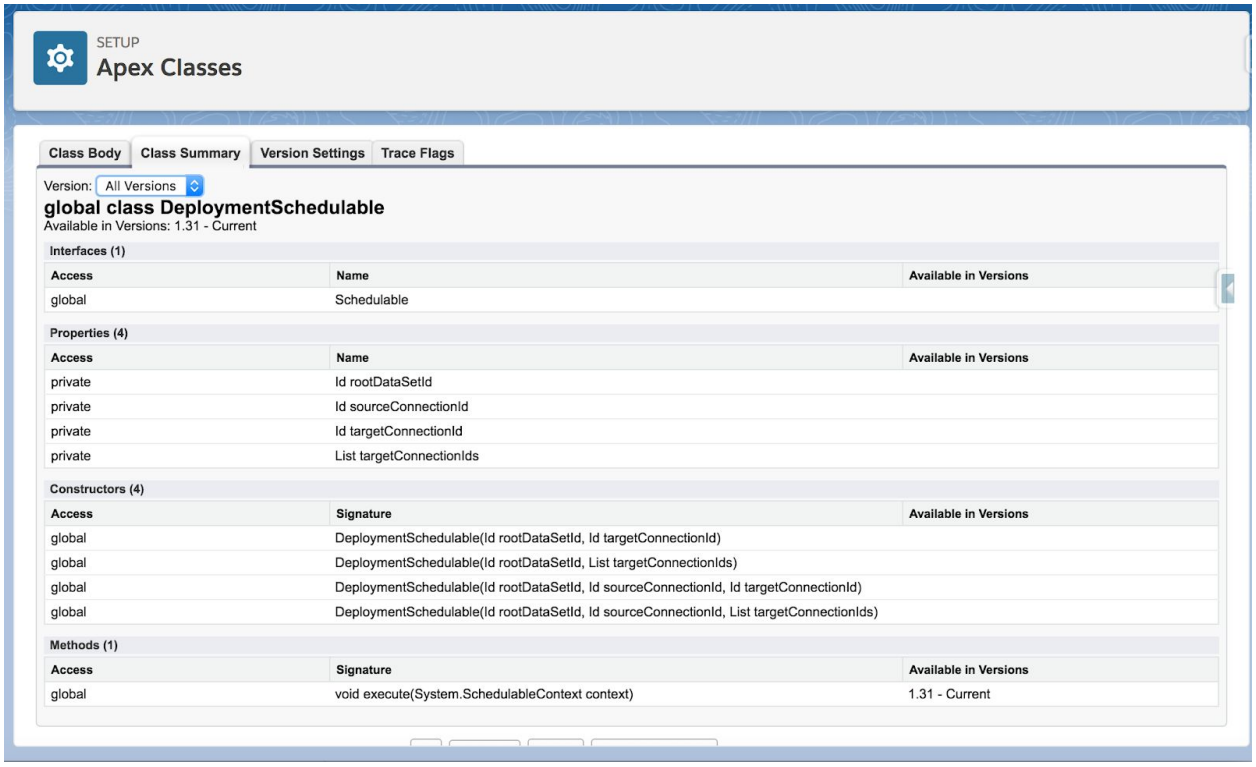
API Error Messages

The following table lists errors the API can return.

Error Code	Error Key	Error Message
209	INVALID_DEPLOYMENT_PLAN_ID	Invalid deployment plan id <deploymentPlanId> in API request.
210	INVALID_DATA_SET_ID	Invalid data set id <dataSetId> in API request.
211	INVALID_SOURCE_CONNECTION	Invalid source org connection id <sourceConnectionId> in API request.
212	INVALID_DESTINATION_CONNECTION	Invalid destination org connection id <destinationConnectionId> in API request.
213	TOO_MANY_DESTINATIONS_ORG_IDS	Too many destination orgs. Please limit the number of destination orgs to five per deployment.

Deployment Scheduling

AppOps provides the `class DeploymentSchedulable` Apex wrapper that you can call to schedule your deployments. Pass in the data set ID, and source and destination org IDs. Refer to the following screenshot from **Setup > Custom Code > Apex Classes > Class Summary** for details:



The following steps provide one way to implement deployment scheduling:

1. In the upper right of your org, click your name in Classic or the gear icon in Lightning Experience, and select **Developer Console**.
2. Navigate to **Debug > Open Execute Anonymous Window**.
3. Paste the following code:

```
Id rootDataSetId = [SELECT Id from PDRI__DataSet__c where
Name = '<data set name>' LIMIT 1].Id;
Id targetConnectionId = [SELECT Id from
PDRI__Connection__c where Name = '<connection name>' LIMIT
1].Id;
DeploymentSchedulable deploymentSchedulable = new
DeploymentSchedulable(rootDataSetId, targetConnectionId);

String schedule = '0 27 11 23 3 ? 2018';
String jobId = System.schedule('Data Set Deployment Job',
schedule, deploymentSchedulable);
```

4. Modify the code to use the appropriate constructor (refer to screenshot for choices) and schedule string.
5. Modify *<data set name>* and *<connection name>* in the query strings to match your names.
6. Execute the code.

Salesforce DX Plugin

Prodly provides a Salesforce DX plugin so Salesforce developers can easily seed their scratch orgs with data and/or deploy data from org to org directly from the command line interface. Scratch orgs, like development sandboxes, don't include data when created. Because scratch orgs are typically created and dissolved on a daily basis, quickly and easily adding data to and deploying data from scratch orgs is critical for developer productivity.

The plugin allows you to deploy data between the following org types for any orgs you have credentials to access:

- From a developer hub (dev hub) org to a scratch org.
- From any org to a scratch org.
- From a scratch org to any org, except a dev hub org.
- From any non-scratch org to any non-scratch org.

For example, you can copy a subset of data from an existing production org to a scratch org for testing purposes. Or, after developing and testing an application using scratch orgs, you can deploy the changes to a centralized sandbox. Or, you can use the plugin for many other possibilities including the typical org-to-org deployment you do from the AppOps Release user interface.

Setup

To properly set up your environment for using the plugin:

1. Follow the instructions in the *Salesforce DX Setup Guide* to enable the Salesforce Developer Hub (dev hub) in your production or business org.
2. Follow the instructions in chapter 1 of *Prodly AppOps Release for Salesforce* to install the Prodly AppOps managed package in your dev hub org. The dev hub org must be the AppOps control org.
3. Download the Prodly AppOps Salesforce DX plugin from https://github.com/prodly/appopsdxcli or https://www.npmjs.com/package/mooverdxcli and install the plugin in your dev hub org.
4. In Salesforce, enable the **View Encrypted Data** permission in the system settings on the profile of the user executing the plugin commands. This permission is required to successfully query the control org connection.
5. Use the Salesforce DX command-line interface (CLI) to authorize your dev hub org.
6. For any other existing orgs relevant to your use case (such as a full sandbox), either use the **Connections** tab in the AppOps app to create connections to the orgs or use the Salesforce DX CLI to authorize the orgs.
7. Use the Salesforce DX (sfdx) CLI to create your scratch orgs.

8. Optionally, use the following command to set a default dev hub username:
   ```
   $ sfdx force:config:set
   defaultdevhubusername=<devhubusername_or_alias> --global
   ```
9. Optionally, use the following command to set a default scratch org username:
   ```
   $ sfdx force:config:set defaultusername=<username_or_alias>
   --global
   ```

Usage

To deploy data to or from an org, enter the following command at the command line prompt in your Salesforce DX CLI command shell:

```
$ sfdxMoover:deploy INSTRUCTIONS [orgS] [OPTIONS]
```

The parameters provide for maximum flexibility when using the CLI command. Deployment requires these three pieces of information:

- Deployment instructions (deployment plan or data set)
- The org to use as the source of the data to deploy
- The org to use as the source destination of the data to deploy

The following table lists all the available parameters:

Parameter	Purpose
Deployment Instructions	
`-p <deployment plan>, --plan=<deployment plan>`	Use to specify the name or record ID of the deployment plan to deploy.
`-t <dataset>, --dataset=<dataset>`	Use to specify the name or record ID of the data set to deploy.
Notes: ● Specify exactly one deployment instructions parameter. Specifying zero or more than one deployment instructions parameter results in an error. ● For each data set and deployment plan in the deployment, **Active** must be selected in the corresponding AppOps record.	
Orgs	
None	Use to specify the dev hub org as the source org and the scratch org as the destination org.
`-s connection, --source=<connection>`	Use to specify an AppOps connection name or record ID as the source org and the scratch org as the destination org.

`-d <connection>`, `--destination=<connection>`	Use to specify the scratch org as the source org and an AppOps connection name or record ID as the destination org.
Notes: ● Specify zero, one, or both org parameters. ● When not specifying an org parameter, you must specify the scratch and dev hub orgs either with the optional setup steps or with the `-u/--targetusername` and `-v/--targetdevhubusername` options. ● When specifying an AppOps connection record, **Active** must be selected in the AppOps connection record. ● When specifying an AppOps connection record as the destination org, **Do Not Allow As Destination** must not be selected in the AppOps connection record. ● When specifying an AppOps connection record by a name that exists for more than one record, AppOps uses the last-modified connection. ● The dev hub org cannot be the destination org.	
Options	
`--apiversion=<apiversion>`	Use to override the API version used for API requests made by this command.
`--json`	Use to receive an output response. When included, the response is in JSON format. When not included, no response is given.
`--loglevel=<level>`	Use to specify the logging level for this command invocation with one of the following levels: ● error ● warn ● info ● debug ● trace ● fatal For details, refer to https://developer.salesforce.com/docs/atlas.en-us.sfdx_dev.meta/sfdx_dev/sfdx_dev_cli_log_messages.htm.
`-u <username>`, `--targetusername=<username>`	Use to specify the username or alias of the scratch org, overriding any default scratch org you specified in the optional setup steps.

`-v <username>, --targetdevhubusername=<username>`	Use to specify the username or alias of the dev hub org, overriding any dev hub org you specified in the optional setup steps.

The AppOps service retrieves the active control org connection information from the dev hub control org. If not found, the service returns an error and terminates.

The scratch and dev hub org connections are based on access tokens only. You cannot use refresh tokens. This requirement means the session settings you configure in the orgs apply. Any deployment running for longer than the session timeout terminates on next connection access.

For any deployment involving the dev hub org, the plugin uses the control org connection from AppOps as the control org connection for the actual deployment, not the dev hub org connection from DX. The user who created the control org connection in AppOps can be different than the user who authenticated the dev hub org in Salesforce DX.

The plugin uses the DX dev hub connection to query for connections and insert the result, and as the source connection when the source org is also the control org. The AppOps service uses the control org connection to report results and the source connection to query for data and metadata.

AppOps service inserts deployment result record into your control org. In the AppOps app, monitor the deployment and view the results on the **Deployment Results** tab.

Response

When you include `--json` in your `AppOps:deploy` command, you receive the following output when the deployment initiates:

```
{
        "Deployment launched with the result ID "${insertResult(0).id}
}
```

Examples

Command	Result
$ sfdx AppOps:deploy –p MyDeploymentPlan	Uses a deployment plan to deploy data to the default scratch org from the default dev hub org.
$ sfdx AppOps:deploy –t MyDataSet –u FixesScratchOrg –v MainDevHub	Uses a data set to deploy data to a specified scratch org from a specified dev hub org.

$ sfdx AppOps:deploy --dataset=MyDataSet --targetusername=test-utxac7gbati9@example.com --targetdevhubusername=jsmith@acme.com	Uses a data set to deploy data to a specified scratch org from a specified dev hub org using long parameter names.
$ sfdx AppOps:deploy -t MyDataSet -d "UAT Sandbox Connection"	Uses a data set to deploy data to a specified UAT sandbox org from the default scratch org using the named connection record.
$ sfdx AppOps:deploy --plan=MyDeploymentPlan --targetusername=test-utxac7gbati9@example.com --source "Production Connection"	Uses a deployment plan to deploy data to a specified scratch org from the production org using the named connection record and long parameter names.