

# Forge to APS Migration Guide

Complete Migration Strategy for Modern APS Development

## Migration Overview

From: Autodesk Forge (Legacy)  
To: Autodesk Platform Services (APS)  
Tool: RAPS CLI v4.2.1  
Estimated Time: 2-4 weeks  
Pages: 9

# Contents

<b>1 Executive Summary</b>	<b>3</b>
<b>2 Migration Overview</b>	<b>3</b>
2.1 Key Differences: Forge vs APS . . . . .	3
2.2 Migration Benefits with RAPS . . . . .	3
<b>3 Pre-Migration Assessment</b>	<b>3</b>
3.1 Current Forge Usage Inventory . . . . .	3
3.2 APS App Configuration . . . . .	4
<b>4 RAPS CLI Migration Strategy</b>	<b>4</b>
4.1 Install and Configure RAPS . . . . .	4
4.2 API Endpoint Migration . . . . .	4
4.2.1 Authentication Migration . . . . .	4
4.2.2 Data Management Migration . . . . .	5
4.2.3 Model Derivative Migration . . . . .	5
<b>5 Code Migration Examples</b>	<b>5</b>
5.1 Node.js Application Migration . . . . .	5
5.1.1 Before: Forge SDK Implementation . . . . .	5
5.1.2 After: RAPS Integration . . . . .	6
5.2 Python Application Migration . . . . .	6
5.2.1 Before: Forge SDK Implementation . . . . .	6
5.2.2 After: RAPS Integration . . . . .	7
<b>6 Advanced Migration Scenarios</b>	<b>7</b>
6.1 CI/CD Pipeline Migration . . . . .	7
6.1.1 GitHub Actions Migration . . . . .	7
6.2 Viewer Migration . . . . .	8
6.2.1 Forge Viewer to APS Viewer . . . . .	8
<b>7 Testing and Validation</b>	<b>8</b>
7.1 Migration Testing Strategy . . . . .	8
7.2 Validation Checklist . . . . .	9
<b>8 Go-Live and Monitoring</b>	<b>9</b>
8.1 Deployment Strategy . . . . .	9
8.2 Post-Migration Monitoring . . . . .	9
<b>9 Migration Timeline</b>	<b>9</b>

## 1 Executive Summary

This guide provides a comprehensive migration path from legacy Autodesk Forge APIs to modern Autodesk Platform Services (APS) using RAPS CLI for automation and efficiency.

### Important Notice

#### Forge API End-of-Life Timeline:

- **March 2025:** New Forge registrations disabled
- **September 2025:** Forge APIs deprecated
- **March 2026:** Forge APIs sunset (end-of-life)

## 2 Migration Overview

### 2.1 Key Differences: Forge vs APS

rapslight Aspect	Forge (Legacy)	APS (Modern)
Base URL	developer.api.autodesk.com	developer.api.autodesk.com
Authentication	OAuth 2.0	OAuth 2.0 (Enhanced)
SDK Libraries	forge-apis	aps-sdk-*
Documentation	forge.autodesk.com	aps.autodesk.com
Viewer	Forge Viewer	APS Viewer
Model Derivative	/modelderivative/v2	/modelderivative/v2
Data Management	/project/v1	/project/v1
Object Storage	/oss/v2	/oss/v2

### 2.2 Migration Benefits with RAPS

- **Automated Migration:** RAPS handles API endpoint transitions
- **Unified CLI:** Single tool for all APS operations
- **Modern Tooling:** Enhanced error handling and retry logic
- **CI/CD Ready:** Built for automation and DevOps workflows
- **Future-Proof:** Active development and APS alignment

## 3 Pre-Migration Assessment

### 3.1 Current Forge Usage Inventory

#### Step 1: Document Current Implementation

```
# Audit current Forge usage
grep -r "developer.api.autodesk.com" ./src
grep -r "forge-apis" ./package.json
grep -r "Autodesk.Viewing" ./src
```

### Migration Checklist

- Authentication flows (2-legged, 3-legged)
- API endpoints used (DM, OSS, Model Derivative)
- Forge Viewer implementations
- Webhook configurations
- SDK dependencies
- Custom integrations

## 3.2 APS App Configuration

### Step 2: Create APS Application

1. Navigate to APS Developer Portal
2. Create new application
3. Configure OAuth settings
4. Note Client ID and Secret
5. Enable required APIs

## 4 RAPS CLI Migration Strategy

### 4.1 Install and Configure RAPS

#### Step 3: RAPS Setup

```
# Install RAPS CLI
brew install autodesk-platform-services/tap/raps

# Configure APS credentials
raps profile create migration-project \
--client-id YOUR_APS_CLIENT_ID \
--client-secret YOUR_APS_CLIENT_SECRET \
--scopes "data:read,data:write,viewables:read"

# Test connection
raps auth client-credentials
raps dm hubs
```

### 4.2 API Endpoint Migration

#### Step 4: Update API Calls

##### 4.2.1 Authentication Migration

rapslight Forge Implementation	RAPS Migration
<pre>const AuthClientTwoLegged = require('forge-apis').AuthClientTwoLegged;  const autoRefresh = true; const oAuth2TwoLegged = new AuthClientTwoLegged(   CLIENT_ID,   CLIENT_SECRET,   ['data:read'],   autoRefresh );</pre>	<pre># RAPS handles authentication \ raps auth client-credentials \ --scopes "data:read"  # Use in scripts export APS_TOKEN=\$(raps auth token)</pre>

#### 4.2.2 Data Management Migration

rapslight Forge Implementation	RAPS Migration
<pre>const projectApi = new ProjectApi();  projectApi.getHubs(   oauth2client,   opts ).then((data) =&gt; {   console.log(data.body.data); });</pre>	<pre># Direct command raps dm hubs --output json  # In scripts HUBS=\$(raps dm hubs --output json) echo \$HUBS   jq '.data[].id'</pre>

#### 4.2.3 Model Derivative Migration

rapslight Forge Implementation	RAPS Migration
<pre>const derivativeApi = new DerivativesApi();  derivativeApi.translate(   translateJob,   opts,   oauth2client ).then((data) =&gt; {   console.log(data.body.urn); });</pre>	<pre># Start translation raps translate \$URN \ --output-formats svf2  # Check status raps translate status \$URN  # Download derivatives raps translate download \$URN</pre>

## 5 Code Migration Examples

### 5.1 Node.js Application Migration

#### Step 5: Refactor Application Code

##### 5.1.1 Before: Forge SDK Implementation

```
// package.json dependencies
"forge-apis": "^0.9.7"
```

```
// app.js
const forgeSDK = require('forge-apis');

async function uploadFile(bucketKey, objectName, filePath) {
  const oauth = new forgeSDK.AuthClientTwoLegged(
    process.env.FORGE_CLIENT_ID,
    process.env.FORGE_CLIENT_SECRET,
    ['bucket:create', 'bucket:read', 'data:read', 'data:write']
  );

  await oauth.authenticate();

  const ossApi = new forgeSDK.ObjectsApi();
  const uploadResult = await ossApi.uploadObject(
    bucketKey,
    objectName,
    fs.createReadStream(filePath).byteLength,
    fs.createReadStream(filePath),
    {},
    oauth,
    oauth.getCredentials()
  );

  return uploadResult.body.objectId;
}
```

### 5.1.2 After: RAPS Integration

```
// package.json - no SDK dependencies needed
// Use RAPS CLI via child_process

const { execSync } = require('child_process');

async function uploadFile(bucketKey, objectName, filePath) {
  // RAPS handles authentication automatically
  const command = `raps oss upload ${bucketKey} ${filePath} --object-name ${objectName}
  } --output json`;

  try {
    const result = execSync(command, { encoding: 'utf8' });
    const uploadResult = JSON.parse(result);
    return uploadResult.objectId;
  } catch (error) {
    throw new Error(`Upload failed: ${error.message}`);
  }
}
```

## 5.2 Python Application Migration

### 5.2.1 Before: Forge SDK Implementation

```
# requirements.txt
forge-python-wrapper==1.0.0

# app.py
import forge

def get_token():
    client_id = os.environ['FORGE_CLIENT_ID']
    client_secret = os.environ['FORGE_CLIENT_SECRET']

    auth = forge.Auth(client_id, client_secret)
    token = auth.get_token(['data:read'])
    return token['access_token']
```

```
def list_hubs(token):
    dm = forge.DataManagement(token)
    hubs = dm.get_hubs()
    return hubs['data']
```

### 5.2.2 After: RAPS Integration

```
# requirements.txt - no SDK dependencies needed

import subprocess
import json

def list_hubs():
    # RAPS handles authentication
    result = subprocess.run(
        ['raps', 'dm', 'hubs', '--output', 'json'],
        capture_output=True,
        text=True,
        check=True
    )

    hubs_data = json.loads(result.stdout)
    return hubs_data['data']
```

## 6 Advanced Migration Scenarios

### 6.1 CI/CD Pipeline Migration

#### Step 6: Update Automation Workflows

##### 6.1.1 GitHub Actions Migration

```
# .github/workflows/deploy.yml - Before (Forge)
name: Deploy with Forge
jobs:
  deploy:
    steps:
      - name: Setup Node.js
        uses: actions/setup-node@v3
        with:
          node-version: '16'

      - name: Install Forge SDK
        run: npm install forge-apis

      - name: Upload Files
        env:
          FORGE_CLIENT_ID: ${{ secrets.FORGE_CLIENT_ID }}
          FORGE_CLIENT_SECRET: ${{ secrets.FORGE_CLIENT_SECRET }}
        run: node upload-script.js
```

```
# .github/workflows/deploy.yml - After (RAPS)
name: Deploy with RAPS
jobs:
  deploy:
    steps:
      - name: Install RAPS
        run:
          curl -fsSL https://get.rapscli.xyz | bash
          echo "$HOME/.raps/bin" >> $GITHUB_PATH
```

```

- name: Deploy with RAPS
  env:
    APS_CLIENT_ID: ${secrets.APS_CLIENT_ID}
    APS_CLIENT_SECRET: ${secrets.APS_CLIENT_SECRET}
  run:
    - raps auth client-credentials
    - raps oss upload mybucket ./models/*.rvt
    - raps translate --batch ./models/

```

## 6.2 Viewer Migration

### Step 7: Update Viewer Implementation

#### 6.2.1 Forge Viewer to APS Viewer

```

<!-- Before: Forge Viewer -->
<script src="https://developer.api.autodesk.com/modelderivative/v2/viewers/7.*/
  viewer3D.min.js"></script>
<link rel="stylesheet" href="https://developer.api.autodesk.com/modelderivative/v2/
  viewers/7.*/style.min.css">

<script>
var viewer = new Autodesk.Viewing.GuiViewer3D(viewerDiv);
Autodesk.Viewing.Initializer(options, function onInitialized() {
  viewer.start(svfUrl, loadOptions, onSuccess, onError);
});
</script>

<!-- After: APS Viewer -->
<script src="https://developer.api.autodesk.com/modelderivative/v2/viewers/viewer3D.
  min.js"></script>
<link rel="stylesheet" href="https://developer.api.autodesk.com/modelderivative/v2/
  viewers/style.min.css">

<script>
// Same API - minimal changes needed
var viewer = new Autodesk.Viewing.GuiViewer3D(viewerDiv);
Autodesk.Viewing.Initializer(options, function onInitialized() {
  viewer.start(svfUrl, loadOptions, onSuccess, onError);
});
</script>

```

## 7 Testing and Validation

### 7.1 Migration Testing Strategy

#### Step 8: Comprehensive Testing

##### 1. Authentication Testing

```

raps auth status --verbose
raps auth token --decode

```

##### 2. API Functionality Testing

```

# Test each migrated workflow
raps dm hubs
raps dm projects HUB_ID
raps oss buckets
raps translate URN

```

##### 3. Performance Comparison

```
# Benchmark migration performance
time raps dm hubs
time raps translate large-model.rvt
```

## 7.2 Validation Checklist

### Migration Validation

- All Forge API calls replaced with RAPS
- Authentication flows working
- File upload/download functional
- Translation workflows operational
- Viewer integration updated
- Error handling implemented
- Performance metrics acceptable
- CI/CD pipelines functional

## 8 Go-Live and Monitoring

### 8.1 Deployment Strategy

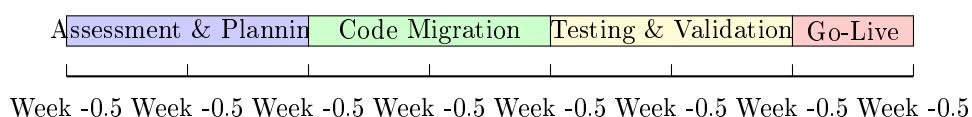
#### Step 9: Production Deployment

1. **Staged Rollout:** Deploy to staging environment first
2. **Parallel Operation:** Run both systems during transition
3. **Monitoring Setup:** Implement comprehensive logging
4. **Rollback Plan:** Prepare fallback procedures
5. **User Communication:** Notify stakeholders of changes

### 8.2 Post-Migration Monitoring

```
# Monitor RAPS operations
raps logs --level error --last 24h
raps stats --api-usage
raps health check
```

## 9 Migration Timeline



### Migration Support

migration-support@rapscli.xyz | Discord: discord.gg/raps

Professional Migration Services Available | rapscli.xyz/migration