

Streamline GIS Content Management with ArcGIS[®] API for Python[®]

Official Esri Training Courseware



esri[®]

THE
SCIENCE
OF
WHERE[®]

Streamline GIS Content Management with ArcGIS[®] API for Python[®]

STUDENT EDITION

Copyright © 2025 Esri

All rights reserved.

Course version 1.0. Version release date September 2025.

Printed in the United States of America.

The information contained in this document is the exclusive property of Esri. This work is protected under United States copyright law and other international copyright treaties and conventions. No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as expressly permitted in writing by Esri. All requests should be sent to Attention: Director, Contracts and Legal, Esri, 380 New York Street, Redlands, CA 92373-8100, USA.

Export Notice: Use of these Materials is subject to U.S. export control laws and regulations including the U.S. Department of Commerce Export Administration Regulations (EAR). Diversion of these Materials contrary to U.S. law is prohibited.

The information contained in this document is subject to change without notice.

Commercial Training Course Agreement Terms: The Training Course and any software, documentation, course materials or data delivered with the Training Course is subject to the terms of the Master Agreement for Products and Services found at the following website: esri.com/TrainingTerms. The license rights in the Master Agreement strictly govern Licensee's use, reproduction, or disclosure of the software, documentation, course materials and data. Training Course students may use the course materials for their personal use and may not copy or redistribute for any purpose. Contractor/Manufacturer is Esri, 380 New York Street, Redlands, CA 92373-8100, USA.

Esri Marks: Esri marks and product names mentioned herein are subject to the terms of use found at the following website: esri.com/EsriMarks.

Other companies and products or services mentioned herein may be trademarks, service marks, or registered marks of their respective mark owners.

Table of Contents

Esri resources for your organization	iv
--	----

Course introduction

Course introduction	1
Course goals	1
Icons used in this workbook	2

1 Connecting to an organization

Lesson introduction	1-1
The role of authentication.....	1-2
Using the GIS class	1-5
Authentication schemes	1-7
Explore authentication resources	1-10
Exercise 1: Connect to an organization	1-11
Open ArcGIS Pro and sign in to an organization	1-12
Test anonymous access.....	1-12
Authenticate an organization	1-14
Test authentication using ArcGIS Pro	1-15
Python libraries	1-18
Lesson review	1-21
Answers to lesson 1 questions.....	1-22

2 Strategies for content management

Lesson introduction	2-1
Content management tasks	2-2
Content management workflows	2-3
Explore the ContentManager class	2-7
Using the ContentManager class.....	2-9
Exercise 2: Apply strategies for managing content	2-10
Connect to your organization	2-11
Access an item	2-11
Examine item details	2-12

Access multiple items.....	2-14
Update item details	2-16
Lesson review	2-18
Answers to lesson 2 questions.....	2-19

3 Workflows for feature layer management

Lesson introduction	3-1
Workflows for feature layer management	3-2
Explore the features module	3-4
Accessing feature layer properties	3-5
Exercise 3: Manage feature layer content	3-6
Access a feature service	3-7
Examine feature layer properties.....	3-9
Verify field information for many layers	3-11
Lesson review	3-13
Answers to lesson 3 questions.....	3-14

4 Strategies for updating feature layer properties

Lesson introduction	4-1
Strategies for updating feature layer properties	4-2
Explore update capabilities in the FeatureLayer class	4-5
Modifying a feature layer definition	4-6
Exercise 4: Update feature layer properties.....	4-7
Add a status field to a single service	4-8
Add a status field to multiple services	4-9
Change the layer name within the feature collection	4-10
Lesson review	4-14
Answers to lesson 4 questions.....	4-15

5 Strategies for editing feature layers

Lesson introduction	5-1
Strategies for editing feature layers	5-2
Explore edit capabilities in the FeatureLayer class.....	5-6
Feature Service data structures	5-7

Exercise 5: Edit feature layers	5-8
Engineer input data.....	5-9
Add features.....	5-11
Update features	5-12
Delete features.....	5-15
Alternative strategies to edit feature layers.....	5-17
Lesson review	5-18
Answers to lesson 5 questions.....	5-19

Appendices

Appendix A: Data license agreement.....	A-1
Appendix B: Answers to lesson review questions.....	B-1
Appendix C: Additional resources	C-1
Appendix D: Lesson stories.....	D-0

Esri resources

Take advantage of these resources to develop ArcGIS® software skills, discover applications of geospatial technology, and tap into the experience and knowledge of the ArcGIS community.

Instructor-led and e-Learning resources

Esri® instructor-led courses and e-Learning resources help you develop and apply ArcGIS skills, recommended workflows, and best practices. View all training options at **esri.com/training/catalog/search**.

Planning for organizations

Esri training consultants partner with organizations to provide course recommendations for job roles, short-term training plans, and workforce development plans. Contact an Esri training consultant at **training@esri.com**.

Esri technical certification

The Esri Technical Certification Program recognizes qualified individuals who are proficient in best practices for using Esri software. Esri exams are designed to validate expertise with ArcGIS products, such as ArcGIS Online, ArcGIS Pro and extensions, ArcGIS Enterprise and extensions, and developer technologies. Certifications are offered at three levels: foundation, associate, and professional. Learn more at **esri.com/training/certification**.

Social media and publications

X (formerly Twitter): **[@EsriTraining](https://twitter.com/EsriTraining) and [@Esri](https://twitter.com/Esri)**

Esri on LinkedIn: **linkedin.com/company/esri**

Esri training blog: **esri.com/trainingblog**

Esri publications: Access online editions of ArcNews, ArcUser, and ArcWatch at **esri.com/esri-news/publications**

Esri training newsletter: Subscribe at **go.esri.com/training-news**

Other Esri newsletters: Subscribe to industry-specific newsletters at **go.esri.com/subscribe**

Esri resources (continued)

Esri Press

Esri Press publishes books on the science and technology of GIS in numerous public and private sectors. **esripress.esri.com**

GIS bibliography

A comprehensive index of journals, conference proceedings, books, and reports related to GIS, including references and full-text materials. **gis.library.esri.com**

ArcGIS documentation and tutorials

In-depth information, tutorials, and documentation for ArcGIS products.

ArcGIS Online: **arcgis.com**

ArcGIS Desktop: **desktop.arcgis.com**

ArcGIS Enterprise: **enterprise.arcgis.com**

ArcGIS Pro: **pro.arcgis.com**

Esri Community

Join the online community of GIS users and experts. **community.esri.com**

Esri events

Esri conferences and user group meetings offer a great way to network and learn how to achieve results with ArcGIS. **esri.com/events**

Esri videos

View an extensive collection of videos by Esri leaders, event keynote speakers, and product experts. **mediaspace.esri.com**

Esri resources (continued)

ArcGIS for Personal Use

Improve your GIS skills at home and use ArcGIS to enhance your personal projects. The ArcGIS for Personal Use program includes a 12-month term license for ArcGIS Pro, extension products, and an ArcGIS Online named user account with 100 service credits. **esri.com/personaluse**

GIS Dictionary

This term browser defines and describes thousands of GIS terms. **support.esri.com/en-us/gis-dictionary**

Course introduction

The ArcGIS API for Python helps users automate a variety of workflows within ArcGIS Online or ArcGIS Enterprise. Understanding the functions and resources that are part of the Python API can help someone automate a wide range of tasks.

This course will introduce how the Python API can be applied to a range of content access and management tasks. You will use documentation and samples to gain an understanding of how to best to automate a task with a script. You will apply a stepped technique for accessing and modifying items, as well as learning to edit features within an organization.

Course goals

After completing this course, you will be able to perform the following tasks:

- Understand different authentication schemes to connect to an organization.
- Learn strategies for accessing and managing content using manager subclasses.
- Evaluate workflows for feature layer management.
- Use different update strategies to update feature layers.

Icons used in this workbook



Estimated times provide guidance on approximately how many minutes an exercise will take to complete.



Notes indicate additional information, exceptions, or special circumstances about specific course topics.



Recommended practices improve efficiency and save time.



Esri Academy resources provide more in-depth training on related topics.



Additional resources provide additional information about related topics.



Warnings indicate potential problems or actions that should be avoided.

1

Connecting to an organization

When you access an organization through ArcGIS Online or ArcGIS Enterprise, you are required to authenticate who you are. When writing code using the ArcGIS API for Python, you will need to think about this same process. Usually, the first step in writing code with the Python API is connecting to the organization where the task will be performed.

Topics covered

- The role of authentication

- Choosing an authentication scheme

- Using the GIS class to authenticate to an organization

- Python libraries in ArcGIS

The role of authentication

When creating a script to automate a workflow with an organization, it is best to think in terms of what it should accomplish. This process can then be broken down into different steps that will help construct what the code will do at the right moments.

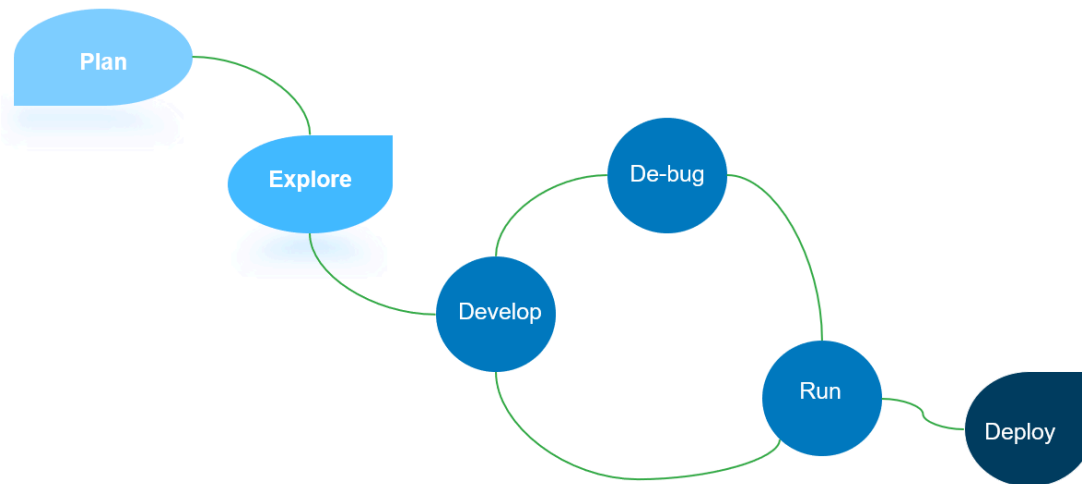


Figure 1.1. You will follow certain workflow steps to create code to automate tasks with the API.

Script creation workflow

The first step is to consider how the code will connect to the organization. If you are familiar with an organization, then you know that there are different roles and user types assigned for those that use the organization. The code that you write has to take that into consideration when it is run.

This simple, four-step workflow provides a good approach to writing your code:

- 1. Plan:** Who will run the code? What will the script accomplish? For which user type will the script connect to the organization? On what OS will the code be run? Are credit-consuming tasks used?
- 2. Explore:** Use resources to help put together the functionality that the script will accomplish.
- 3. Develop/Run/De-bug:** Build, test, and repair code with the resources and tools to bring the workflow to life.

The role of authentication (continued)

4. Deploy: Run the code in the applicable run environment decided upon in the original plan.

When writing code with the ArcGIS API for Python, an important step in the process will be to authenticate to an organization through the GIS class. When you go through the planning phase for your code, it is important to identify which authentication method will be used by the GIS class to access the organization. After the authentication takes place, the code can proceed to the task.



Figure 1.2. There are high-level steps for accessing an organization.



Figure 1.3. There are five organization authentication options within the API.

When you connect to an organization using the ArcGIS API for Python, a similar authentication process is used and is like connecting with a browser. A URL and credentials are needed to gain

The role of authentication (continued)

access to the organization. ArcGIS API for Python code will need to connect to an organization using configured security settings defined by an organization administrator.

Structurally within the Python API, the GIS class is used to facilitate the authentication through various parameters that are passed when instantiated. The parameters that are passed depend on the security method used to sign in to the organization. The Python API supports a wide range of authentication methods for both ArcGIS Online and ArcGIS Enterprise.

The most common parameters used in a script to sign in to an organization are the URL, username, and password. In other situations, a `client_id`, certificate, or different authentication credentials might be used.

Using the GIS class

The GIS class plays an integral part in your code when working with the ArcGIS API for Python. It serves as a connection to ArcGIS Online or ArcGIS Enterprise. The class understands how to connect to your organization through the URL parameter that your code provides. The class supports different authentication methods to accommodate the different authentication patterns used by organizations.

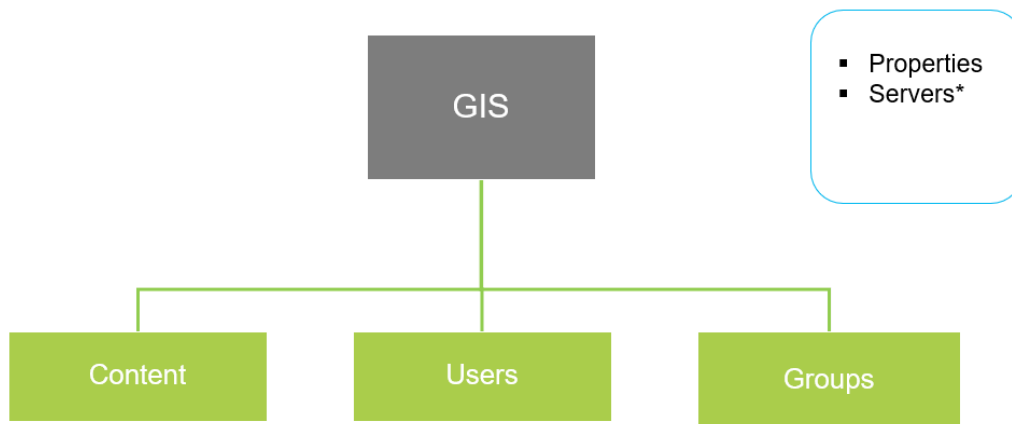


Figure 1.4. The GIS class provides access to content, users, and groups.

The GIS class supports this through the range of parameters your code can use to access an organization. Some of those parameters are listed in the following table.

Parameter name	Description
url	A web address to ArcGIS Online or ArcGIS Enterprise
username	The username to sign in
password	Case-sensitive password for the user
profile	(Optional) Unique machine-based access method

After the connection to an organization is active, the GIS class provides access to a range of properties in an organization. The ability to access these different properties is also dependent on the user type that is associated with the username. That is important because what your code

Using the GIS class (continued)

is capable of is dependent on the user type.

The GIS class provides access to a member's content. Through the content property, you could examine items in the organization. Access to different users and what you can see and do will be based on your user type. The GIS class can help your code access or create different groups within the organization. These examples are just a few capabilities of the GIS class. The Python API reference documentation outlines additional access points.

Authentication schemes

When planning the structure of the code to be written, you should first consider who will be running the code and what access level they have to the organization. Access starts at authentication, and if you or a user of your code does not have the proper credentials, the code may not be able to accomplish its task.

Read the considerations for each authentication method, and then answer the following questions.

Authentication considerations

Authentication	Script considerations	Use case
Anonymous	<ul style="list-style-type: none">• Access an item in the organization that is shared with everyone• Inventory items that are set to this share level and log• Ensure an item that is in the public space has the proper details and metadata• Can connect without credentials	Access a service that has been shared outside of an organization's security and is meant for public use. Use of the service depends on the capabilities that are accessible.

Authentication schemes (continued)

(Table continued from previous page)

Authentication	Script considerations	Use case
Built-in	<ul style="list-style-type: none"> • Apply organization security settings to access protected user, group, and organizational content • User authentication is the most common approach • Align code to the organization authentication scheme 	The script user can apply their credentials to match who they need to be signed in as.
ArcGIS Pro	<ul style="list-style-type: none"> • Access the organization and user type from the ArcGIS Pro session • Use ArcGIS Notebooks and script tools for code options within the desktop environment 	Make sharing a script easier because authentication is through ArcGIS Pro.
Locally stored credentials	<ul style="list-style-type: none"> • Authenticate once and use a profile many times • Profiles may not exist on the machine that is running the code and may produce errors in connecting 	Locally persisted profiles can provide a level of protection to your organization sign-in credentials.

Authentication schemes (continued)

(Table continued from previous page)

Authentication	Script considerations	Use case
OAuth 2.0	<ul style="list-style-type: none">• Have access to everything that you are privileged to access• Access ready-to-use services• Unable to create items	Provide authorization to batch-geocode a file of addresses on your behalf.

1. What should you consider about accessing an organization anonymously?

2. How would using a profile impact a script's access to an organization?

Explore authentication resources

In this activity, you will explore two resources to gain an understanding of how to apply the authentication schemes and additional properties that can be accessed through the class.

Instructions

- a** In a web browser, go to the ArcGIS API for Python documentation: [Working with different authentication schemes](#).
- b** Using the page guide on the right side, explore how the GIS class is used with different authentication methods.
- c** Answer the following questions, and then close the web browser.

1. If you connect anonymously, what can you not do?

2. How long can your code access an organization with ArcGIS Pro closed?

3. What is the recommendation for a noninteractive OAuth sign-in experience?

Connect to an organization

In this exercise, you will authenticate in different ways to different organizations. You will write code to examine the properties that can be accessed after successful authentication has occurred.

In this exercise, you will perform the following tasks:

- Connect to an organization anonymously.
- Write code to examine organization properties.
- Create a persistent profile to authenticate to an organization.

Step 1: Open ArcGIS Pro and sign in to an organization

- a Start ArcGIS Pro.
- b In the ArcGIS Sign in dialog box, expand ArcGIS login, if necessary.
- c Type the username and password provided by your instructor, and then click Sign In.
- d Click Open Another Project.
- e In the Open Project dialog box, on the left side, click This PC.
- f On the right, browse to **C:\EsriTraining\ARPY\San Diego County**.
- g Open the San Diego County project.

Step 2: Test anonymous access

In this step, you will access and view properties with ArcGIS Online anonymously.

- a In the Catalog pane, expand the Notebooks folder and open the Exercise01 notebook.
- b Find the first code cell to import the GIS.
- c Run the cell to import the class.
- d In the next code cell, add the following code:

```
gis = GIS()
```
- e In the same code cell, on a new line, type `gis`.
- f Run the code cell.

Hint: For examples, refer to the ArcGIS API for Python documentation: [Working with different authentication schemes](#).

- g Verify that your output matches the following graphic.

```
In [1]: from arcgis.gis import GIS
```

```
In [3]: gis = GIS()
gis
```

```
Out[3]: GIS @ https://www.arcgis.com
```

The previous process would be similar to opening a browser and visiting www.arcgis.com. In the next step, you will examine different properties that you can access without having signed in.

- h In the next code cell, add the following code:

```
gis.properties
```

- i Run the cell and examine the properties.

The properties return a JSON object, which is a dictionary in Python, that describes different aspects of the organization that your code is connected to.

- j Modify the code cell to the following:

```
gis.properties.portalName
```

- k Run the code cell.

1. What is the name of the portal that is returned?



Because no credentials were used when creating the GIS class, the default connection is with an ArcGIS Online anonymous connection.

- l Modify the code cell to the following:

```
gis.properties.helperServices
```

- m Run the code cell.

2. What does the `helperServices` property tell you about the organization?

In this step, you explored accessing an organization anonymously and you examined properties.

Step 3: Authenticate an organization

The notebook is still signed in to ArcGIS Online. A notebook can be signed in to multiple organizations, as long as the variables that are used are distinctly different. Being signed in to multiple organizations can help your code work with different items and sometimes extract, transform, and load (ETL) data between them.

In this step, you will authenticate to your organization using the provided Training Services account.

- a Click the code cell at the bottom of the notebook to select it.
- b Change the cell to a Markdown cell.
- c Add the following text: `### Access the Training Services Organization`
- d Run the cell.
- e Add the following code in the code cell:

```
gis_EdOrg = GIS()
```



For the next action, refer to the username and password that your instructor provided.

- f Inside the parentheses of the GIS class, add the following parameters:

URL	"https://www.arcgis.com"
username	"[provided by instructor]"
password	"[provided by instructor]"

- g In the same code cell, on a new line, add the following code:

```
gis_EdOrg
```

- h Run the code cell.



The completed code should look like the following image, with your username and password included.

```
gis_EdOrg = GIS("https://www.arcgis.com", "https://www.arcgis.com", "1234567890")
gis_EdOrg
```

GIS @ <https://TrainingServices.maps.arcgis.com>

- i In a new code cell, add the following line of code:

```
gis_EdOrg.users.me
```

- j Run the code cell.

The output of the code is a rich representation of your user information. This method is an easy way to verify how you are connected to an organization.

- k In a new code cell, add the following line of code:

```
gis_EdOrg.properties
```

- l Run the code cell.

- m Review the properties of the organization.

Now that you are signed in with your username to the organization, you will see different properties for the GIS class that you did not see when you signed in anonymously.

Step 4: Test authentication using ArcGIS Pro

Next, you will use the credentials that you used to sign in to ArcGIS Pro to access your organization. In this final step, you will explore how to store the credentials in a persistent profile and examine the profile.

- a Click the code cell that connected you to the Training organization.

- b Add the following code inside the parentheses:

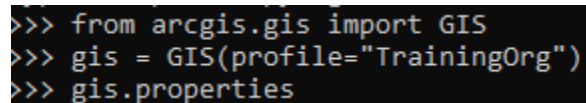
```
profile="TrainingOrg"
```

- c Re-run the code cell.
- d Minimize ArcGIS Pro.

You just created a new profile that can be used on this machine to sign in to the Training Services organization. Next, you will examine the local machine and review the file that is created to store general information about that profile.

- e Click the Windows Start menu and expand ArcGIS.
- f Click Python Command Prompt.
- g Type `python`.
- h Press Enter to activate a Python prompt.
- i Add code to import the GIS class.
- j Create a variable that connects to an organization using the `profile` parameter.
- k Print the properties from the organization.

After completing the imports and connecting to the organization, your command window should look like the following image.



```
>>> from arcgis.gis import GIS
>>> gis = GIS(profile="TrainingOrg")
>>> gis.properties
```

- l Open File Explorer and browse to **C:\Users\Student**.
- m Right-click the `.arcgisprofile` file and choose Edit With Notepad ++.

At the bottom of the file is the final entry showing the profile information used to sign in to the Training Services organization. If this machine is signed in to different organizations, they would be reflected here as other entries. The password information is not stored here; additional credential information needed to sign in to the organization depends on the OS that is used.

For more information about credential managers, go to the ArcGIS API for Python documentation: [Working with different authentication schemes](#).

- n Close both Windows Explorer and the Python Command Prompt.
- o Save your project and leave ArcGIS Pro open.

In this exercise, you explored the GIS class properties and different ways to authenticate to an organization.

Python libraries

Within ArcGIS, there are two main Python libraries that you will use to work with your GIS: the ArcGIS API for Python and ArcPy. The Python API is more closely aligned with helping you automate tasks with ArcGIS Online or ArcGIS Enterprise. ArcPy is more closely associated with automating workflows within ArcGIS Pro.

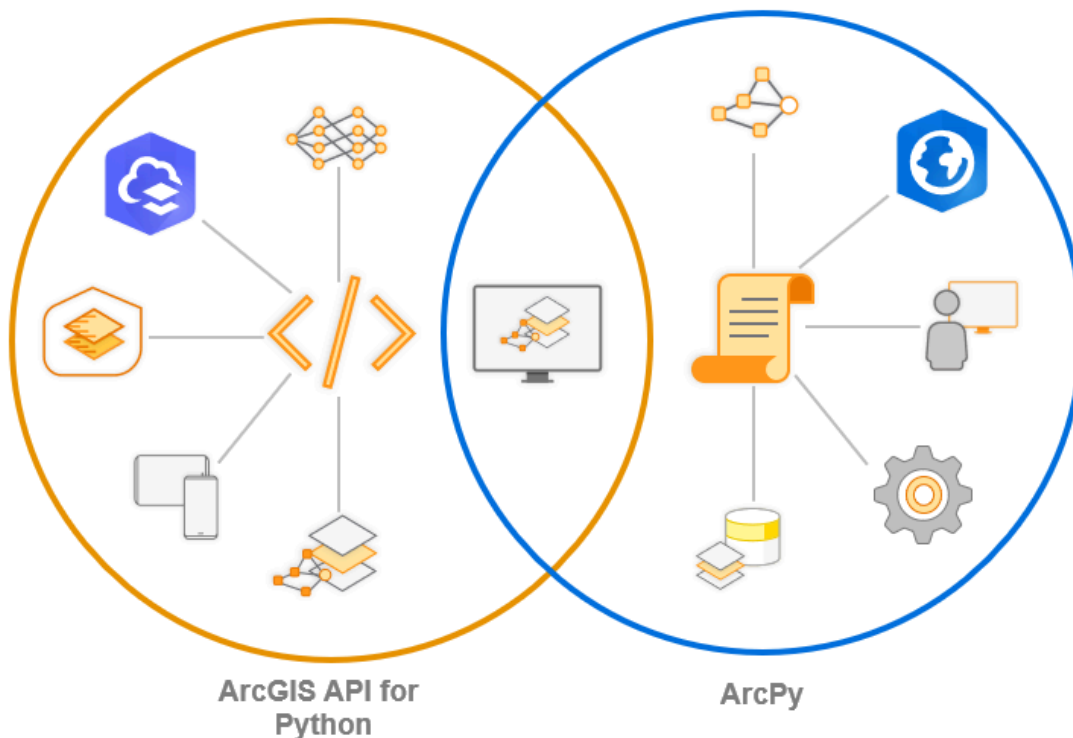


Figure 1.5. The ArcGIS API for Python contains functionality for working with, managing, analyzing, and sharing web content. ArcPy contains functionality for performing ArcGIS Pro tasks, including data management using ArcPy classes and functions, as well as geoprocessing tools.

You can typically accomplish your script's task with everything provided within the Python API. However, there may be times when you need to perform analysis or data access through tools provided by ArcPy.

ArcPy

ArcPy's primary function is to provide access to ArcGIS Pro functionality, enabling automation of ArcGIS Pro workflows. ArcPy has access to all licensed geoprocessing tools and extension tools.

Python libraries (continued)

These geoprocessing tools are focused on analysis, data management, and data conversion workflows. ArcPy also has a variety of submodules that can access data and mapping elements, as well as help automate different publishing routines.

ArcGIS API for Python

The Python API provides automation support in three key areas within ArcGIS Online and ArcGIS Enterprise:

- GIS organization administration
- Content management
- Spatial analysis and data science

Using the Python API, you could automate log examination and parse messages as needed; catalog users and their usage within the organization; and automate the creation of web maps and web apps within the organization.

Using the libraries together

The two ArcGIS Python libraries can work together. The first requirement of using the two together is making sure that you have access and are licensed to use ArcPy. Access to ArcPy is granted through a license, which comes from ArcGIS Pro or ArcGIS Server. Considerations will have to be made about how you and your script user will be accessing this license to run the code.

Within the Python API, there are instances when functionality depends on whether ArcPy is accessible. When working with the `GeoAccessor.from_Featureclass` method, there are optional parameters that can be used that are dependent on whether there is ArcPy access or not. In some situations, additional Python packages can be used to substitute functionality.

Python libraries (continued)



[ArcGIS API for Python documentation: GeoAccessor.from_featureclass](#)

[ArcGIS API for Python documentation: FeatureSet.from_arcpy](#)

[ArcGIS API for Python documentation: GeoAccessor.from_table](#)

[ArcGIS API for Python documentation: GeoAccessor.select](#)

Lesson review

1. What is accomplished by the following code?

```
from arcgis.gis import GIS
enterprise = GIS("https://ebase.ad.local/portal",
profile='MyEnterLogin', verify_cert=False)
```

2. What is accomplished by the following code?

```
from arcgis.gis import GIS
gis = GIS("https://www.arcgis.com", profile='ProductionOrg',
set_active=True)
```

Answers to lesson 1 questions

Authentication schemes (page 1-7)

1. What should you consider about accessing an organization anonymously?

Who will run the script and what the script needs to access

2. How would using a profile impact a script's access to an organization?

Credentials are not shared as part of the script; however, the user running the script may not have that profile set.

Explore authentication resources (page 1-10)

1. If you connect anonymously, what can you not do?

Create or modify content, or perform any analysis tasks

2. How long can your code access an organization with ArcGIS Pro closed?

ArcGIS Pro can remain closed for two weeks by default before the portal token expires.

3. What is the recommendation for a noninteractive OAuth sign-in experience?

Use the built-in identity provider instead of SAML.

Exercise 1: Connect to an organization (page 1-11)

1. What is the name of the portal that is returned?

ArcGIS Online

2. What does the `helperServices` property tell you about the organization?

It shows the URLs of the services that help widgets and applications in an organization.

The ArcGIS API for Python allows you to manage all types of content. Different content managers are used to access maps, layers, users, groups, and resources within your scripts. In this lesson, you will explore and use different resources to access and use these content managers.

Topics covered

- Accessing organization content with the ContentManager class

- Updating content with the ContentManager class

Content management tasks

When using ArcGIS Online or ArcGIS Enterprise, the ArcGIS API for Python is suited to handle a wide variety of tasks. Whether updating the item description for an item or adding one field to 20 different feature services, identifying the potential for automation is a great starting place. In this lesson, you will explore resources to identify possibilities.

What are some examples of content management tasks that you intend to automate with the Python API?

Content management workflows

Whether you are a geospatial analyst, administrator, or developer, you will need to access and connect to content in your organization. When you are writing code with the ArcGIS API for Python, a logical next step after authentication is to seek out the content to connect to for the task that you need to perform.

Manager classes

Working with content through the Python API can bring about different strategies. It is good to focus on the overall goal of the code that can help define how and what content needs to be accessed. After you connect to an organization, the ArcGIS API for Python uses different resource manager classes to enable you to access a variety of content based on the task of the script. These resource manager classes provide access to different properties and methods and can help you access either a single item or a larger number of items, depending on the need of the script.

Content management workflows (continued)

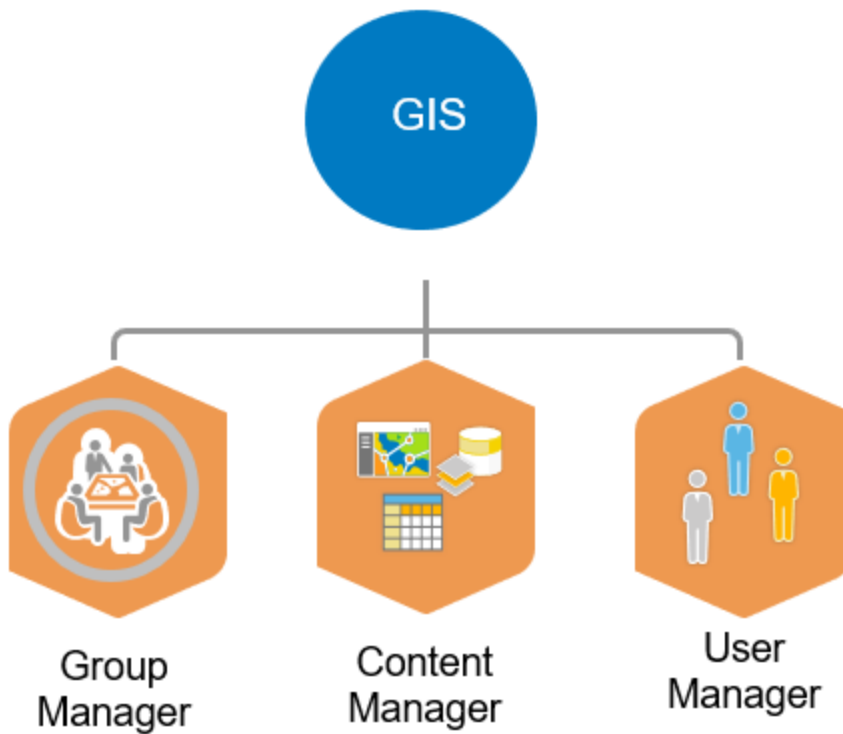


Figure 2.1. The ArcGIS API for Python includes several content manager classes.

Accessing methods

When you need to access a feature layer or web map, you will use the content property available from the GIS class. That content property provides direct access to the ContentManager class within the Python API. The ContentManager class has methods that support accessing, updating, creating, and deleting content.

Content management workflows (continued)

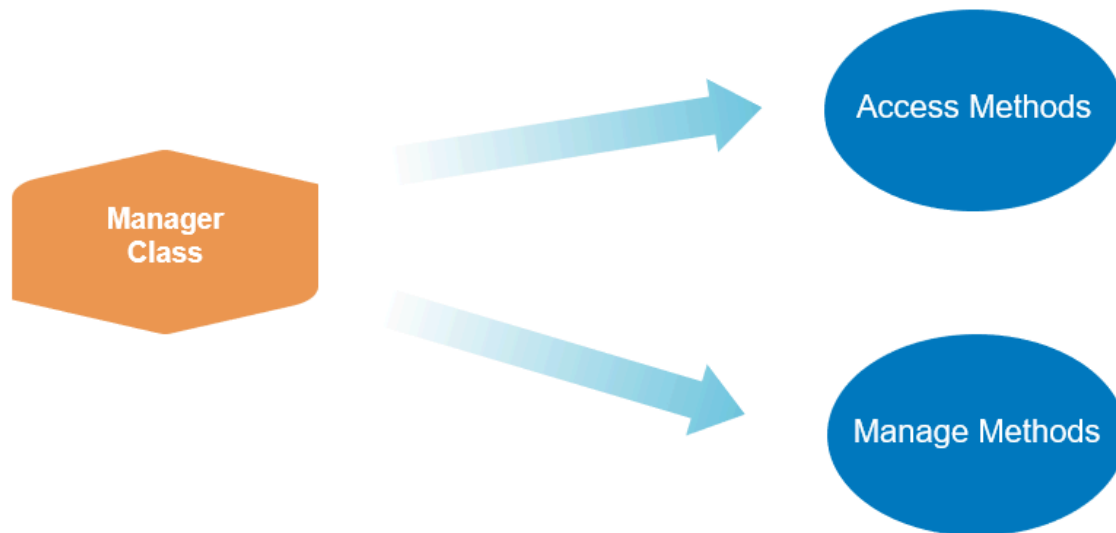


Figure 2.2. The ContentManager class contains various access and modification methods.

The ContentManager class supports various methods that allow you to gain access to different items. Which method is best depends on the task. Sometimes you are updating a single item, and other times you might be working with a variety of items. The following table summarizes the differences.

Method	Description
get	Easy to access an individual item when the itemid is known
search	Similar to organization search and supports a wide number of parameters
advanced_search	Similar to search, supports paging and spatial extent parameters

Managing methods

After you have access to items, the ContentManager class supports methods to modify or administer items within your organization. Which method you use will depend on the overall task of the script. The following table summarizes the differences.

Content management workflows (continued)

Methods	Description
<code>add</code>	Creates a new item in the organization
<code>can_delete</code>	Determines whether an item can be deleted from an organization
<code>clone_items</code>	Allows the ability to clone items from one organization to another
<code>create_folder</code>	Creates a folder to help organize content in an organization
<code>delete_items</code>	Provides the ability to delete a list of items from an organization



[ArcGIS API for Python documentation: property content](#)

[ArcGIS API for Python documentation: ContentManager](#)

[ArcGIS API for Python documentation: Accessing and creating content](#)

[ArcGIS API for Python documentation: Managing your content](#)

Explore the ContentManager class

Throughout the ArcGIS API for Python and API reference documentation, you will find useful samples and conceptual information to help you with your scripts. In this activity, you will explore the documentation to answer questions about accessing content in the ArcGIS API for Python.

Instructions

- a** Open a web browser and go to the ArcGIS API for Python documentation: [ContentManager class](#).
- b** In the menu on the left, expand ContentManager until you see the list of methods.
- c** Review the documentation about the `advanced_search` method to answer the first two questions that follow.
- d** Go to the [Accessing and creating content](#) documentation page.
- e** Under the About Search heading, read the second and third paragraphs, as well as the search tip, to answer the remaining questions.
- f** After you are finished, close the web browser.

1. Which data type can be passed in the `bbox` parameter of the `advanced_search` method?

2. How does the `enrich` parameter change returned search results?

Explore the ContentManager class (continued)

- 3. Does searching for content with the Python API yield similar results to searching with an organization content page?**

- 4. How do Python list comprehensions augment the resource managers' search functions?**

Using the ContentManager class

The ArcGIS API for Python documentation provides valuable contextual information and samples. The contextual information is organized into different capabilities and topics that help you understand how code from the Python API is applied to these topics through samples. The Python API reference documentation helps you understand what you can accomplish by documenting the methods, method parameters, and properties.

The ContentManager class is documented in a number of conceptual help pages, which highlights its importance in the scripting workflow by providing methods to find, create, and access a variety of content in your organization.



ArcGIS API for Python documentation: Accessing and creating content

ArcGIS API for Python documentation: ContentManager class

Apply strategies for managing content

You have learned how to connect to your organization using the Python API. Next, you will continue a workflow that will help you access and manage items within your organization. Imagine that you were recently reviewing various items in your organization and found issues with tags, descriptions, and snippets. You are curious about how often and how many items are affected.

In this exercise, you will perform the following tasks:

- Access items from an organization.
- Update item properties.

Step 1: Connect to your organization

In this step, you will create a new notebook and sign in to your organizational account.

- a Open the ArcGIS Pro project from the previous exercise, if necessary.
- b Verify that ArcGIS Pro is still connected to the Training Services account.
- c Create a new notebook.
- d Rename the notebook to **Exercise02**.
- e Connect to your organization using the authentication steps from the previous exercise.

Hint: Use the URL, username, and password provided by your instructor.

When complete, you should have a `gis` variable that can connect to your organization.

Step 2: Access an item

A colleague of yours indicated that they were using a web map from the organization and saw that various details in the item were either missing or had issues to correct. In this step, you will examine a single item with the Python API to verify.

- a In the Catalog pane, click the Portal tab.
- b Click the Geology of San Diego County web map.
- c In the pop-up, click the URL.

The system's default browser should display and take you to the item page for the geology web map.

- d Sign in with the username and password provided by your instructor, if necessary.
- e On the right side of the item page, expand the Details section, if necessary, and locate the item ID.
- f Select and copy the ID.
- g Return to your Exercise02 notebook in ArcGIS Pro.

- h Paste the ID into a code cell.

Because you are investigating just a single web map, you will use the `get` method to examine a single item.

- i In a web browser, go to the ArcGIS API for Python documentation: [Properties of an item](#).
- j Examine the second cell and how it accesses an item using the `get` method.
- k Copy the code from the page that uses the `get` method.
- l Return to your notebook in ArcGIS Pro.
- m Paste the code into the next code cell in your notebook.
- n Change the variable used for the map to `SD_Geology_Map`.
- o Change the ID to the previously obtained item ID for the web map.

```
SD_Geology_Map = gis.content.get('d7fd78fb56aa48a2a376e08e0c3caefb')
```



Your item ID number will be different from what is shown.

- p To continue adding code on a new line within this cell, press `Enter`.
- q Type `SD_Geology_Map`.
- r Run the current code cell.

The web map is retrieved from the organization, and an enhanced output of the item is displayed within the output of the notebook cell.

Step 3: Examine item details

In this step, you will access different item properties using the notebook cell.

- a Insert a new cell below the active one, if necessary.
- b Add the following code to help access and examine different item properties:

```
SD_Geology_Map.title
```

- c Run the cell and examine the output.
- d Using the same cell and code, examine the following item properties by changing the property and re-running the cell:

- description
- snippet
- tags

Hint: To use autocomplete for the properties, delete the current property, then press Tab for the list of item properties. Use the arrow keys to select the next property and press Enter to add it.

Upon examination, there are some documentation properties missing from this web map. In another case, Diego was misspelled. Next, you will write code to add the proper item documentation for your organization's standards.

- e In a web browser, go to the ArcGIS API for Python documentation: [Item update method](#).
- f Review the `item_properties` parameter data type.

1. Which Python data type is used to update the item properties?

- g Scroll down further to view a small code sample showing how the update process can occur.
- h Return to your notebook in ArcGIS Pro.
- i In a new cell, add the following line of code:
`updated_Properties = {}`
- j Using the provided table, add information to the Python dictionary:

Item property	Value
description	San Diego County regional geology map. This map contains layers to help describe various rock formations throughout all of San Diego County.

(Table continued from previous page)

Item property	Value
snippet	San Diego County Geology map
tags	SANDAG, SD County, Geology

- k** In the same cell, on a new line, add the following code to update the web map item details:
`SD_Geology_Map.update(item_properties=updated_properties)`
- l** Run the cell.
- m** Verify that the changes were made by re-running the cell where the get method accessed the item.

Now that you have corrected a single web map, you will determine whether there are more items that need to be corrected.

Step 4: Access multiple items

In this step, you will use the search method to determine whether there are other items in your organization that have similar issues with the item details and correct them if needed. After finding one item with documentation issues, you want to explore whether there are others with misspellings and missing documentation. You will now examine all the San Diego County web maps in the organization and verify their item information.

- a** In a web browser, go to the ArcGIS API for Python documentation: [Wild card search](#).
- b** Read the example on using a wild card search.

2. Which characters can be used for search character substitution?

- c** In the web browser, highlight the code shown in the following graphic.

```
search_result_USA = gis.content.search(query="title:USA*")
search_result_USA
```


- d Right-click the selected lines and copy the code.
- e In ArcGIS Pro, access the Exercise02 notebook.
- f Insert a new cell in the notebook and paste the code from the browser.
- g In the variable name, replace USA with **SD**.
- h Modify the title query by replacing USA with **SD**.
- i Verify that your code matches the following code:

```
search_result_SD = gis.content.search(query="title:SD*")
search_result_SD
```

- j Run the cell and evaluate the results.

3. Did the query access the correct items, and why or why not?

- k Modify the query as shown in the following lines:

```
search_result_SD = gis.content.search(query="snippet:SanDeigo* AND
type:map AND tags:AUTO")
search_result_SD
```

- l Run the cell and evaluate the results.

Notice that the returned list only contains five web maps. From exploring the organization, you know that there are more. The search method supports a [max_items parameter](#), and its default is 10.

- m Modify the search method as shown in the following lines:

```
search_result_SD = gis.content.search(query="snippet:SanDeigo* AND
type:map AND tags:AUTO", max_items=100)
search_result_SD
```

- n Run the cell and evaluate the results.

Because the snippet was the item detail suspected of having the misspelled value, you will modify the code to verify.

- o At the end of `search_result_SD`, type the following code:

```
[0].snippet
```

- p Re-run the code cell.

Using a variety of parameters in the search method, you identified the different web maps that require updates to their item details.

Step 5: Update item details

Now that you found the different web maps that have the error, you will update the item details for the web maps in the previous step. You will leave the search cell in place so that it can be re-run to identify whether all maps were corrected.

- a Use the Cell tools to create a duplicate of your search code cell.
- b Within the duplicate, add a `for` loop to visit every web map.
- c Inside the loop, print out the `snippet` property of the item.
- d Run the current code cell.



Now that you have found the web maps that need updates, you will apply the changes to multiple maps.

- e Verify that the returned snippets contain the San Diego typo: *SanDeigo*.
- f Open a web browser to the following ArcGIS API for Python documentation pages:
 - [Managing your content](#)
 - [Item update method](#)
- g Using the reviewed resources, modify the loop code to update the web map snippets.
- h Run the cell.

If the process runs successfully, "True" should display below the cell for every web map item that

updates.

To test that the web maps have been modified, you will revisit the first search cell.

-  Click inside the first cell that searches for the web maps.
-  Run the cell.

The cell should display an empty list. If the cell result has web map items in the list, check the list items and review the updated cell code.

-  Save your project and leave ArcGIS Pro open.

Lesson review

1. What is returned by the following code?

```
gis.content.search(query="owner:" + gis.users.me.username,  
                  item_type="Feature Layer",  
                  max_items=15)
```

2. You need to identify feature layers that fall within a new project boundary in your organization. Which method will help you to identify those feature layers?

- a. search
- b. advanced_search
- c. get
- d. analyze

Answers to lesson 2 questions

Content management tasks (page 2-2)

What are some examples of content management tasks that you intend to automate with the Python API?

- Update item descriptions
- Update tags on many items
- Re-assign various items from a deactivated user
- Apply new thumbnails to item descriptions

Explore the ContentManager class (page 2-7)

1. Which data type can be passed in the `bbox` parameter of the `advanced_search` method?

String or list

2. How does the `enrich` parameter change returned search results?

When `enrich` is set to `true`, it will include results that are literal and relevant.

3. Does searching for content with the Python API yield similar results to searching with an organization content page?

No. Other search options work differently than the search that is performed by the ArcGIS API for Python resource managers.

4. How do Python list comprehensions augment the resource managers' search functions?

You can add conditions to the comprehension to further isolate the content being searched for.

Exercise 2: Apply strategies for managing content (page 2-10)

1. Which Python data type is used to update the item properties?

Python dictionary

Answers to lesson 2 questions (continued)

2. Which characters can be used for search character substitution?

Asterisk and question mark

3. Did the query access the correct items, and why or why not?

No. The query accessed all item types instead of just web maps.

3

Workflows for feature layer management

In the previous lesson, you explored ways to access and search for content in your organization. This lesson will focus on one type of content: feature services. You will continue to learn how to access items. You will also learn how to access detailed information about layers and service properties.

Topics covered

- Understanding feature service structure with the Python API

- Accessing feature service properties

Workflows for feature layer management

Feature layers play an integral part within an organization, providing applications with the latest information through updates and edits. For example, a feature layer may have been published from ArcGIS Pro and used in interactive data-driven experiences to keep the public up to date on citywide projects.

In an earlier lesson, you learned how to access different items within an organization. You accessed an item of a particular type (for example, a web map or feature layer) to return item-level properties. Getting item-specific information required passing the item to different API class types.

There are steps to take when accessing a feature layer within your organization. Each step will give you access to different types of information associated with the item or layer. The access workflow provides access to other feature layer management tasks and workflows:

- Search or get returns the item from the organization.
- The item type returned is a feature layer collection.
- The feature layer collection supports many properties, including a layers property.
- Individual feature layers are accessed through the layers property.
- Feature layers can be examined for support of different functions.

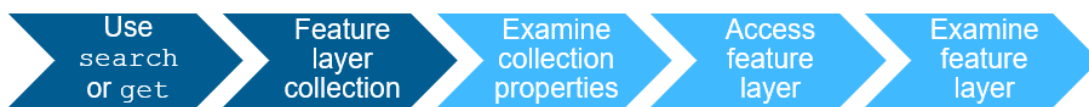


Figure 3.1. You can perform five steps in this workflow.

Feature layer collections

A feature layer collection is a container for one or more feature layers. When not accessed through code, you see it as a feature service REST endpoint or a feature layer item type. A feature layer collection will support a layers or tables property that can be used to access the individual feature layers or tables contained within the service. The feature collection can provide useful information to your script, including the following information:

Workflows for feature layer management (continued)

- Spatial reference
- Geographic data coverage extents
- Supported operations of the services
- Access to layers and tables

Feature layers

A feature layer is made up of a single geometry type with attributes, and it contains features. A feature layer is what some people may think of when adding data to a web map; it is the operational aspect of the service. What a feature layer is capable of within the Python API depends on the properties associated with the service, as defined by the service administrator. With access to a feature layer in the Python API, a script can accomplish the following tasks:

- Answer attribute and spatial questions from the data.
- Support geometry and attribute edits if enabled.

Explore the features module

In this activity, you will use resources to understand how to access properties of feature layers. You will examine samples from the conceptual help and classes from the Python API reference that will help to access feature layer properties.

Instructions

- a In a web browser, go to the ArcGIS API for Python documentation: [Properties of FeatureLayer](#).
- b Review the code examples and statements on accessing properties from a feature layer, and then answer the first two questions that follow.
- c Go to the [FeatureLayer class](#) documentation and review the content for `property` container and `get_unique_values`.
- d Examine the two properties to answer the remaining questions.
- e After you are finished, close the browser.

1. What is returned with `FeatureLayer` properties?

2. How can the properties be accessed?

3. What is returned from `get_unique_values` on the `FeatureLayer` class?

4. Which API class can be accessed through the `container` property?

Accessing feature layer properties

Feature layers are integral to ArcGIS Online and ArcGIS Enterprise deployment. Being able to access and use feature layers in an automated way can make the management of these layers easier. Using similar workflows to those for accessing content, you can access individual layers from a feature layer collection. After you have access to those layers, examining their properties will help you gain insight into how they need to be managed in an automated way.

Manage feature layer content

Accessing property information from layers can be helpful in an organization that can contain a large number of layers. You may be asked to write code using the ArcGIS API for Python that will help examine a range of property types and capabilities from these layers.

In this exercise, you will perform the following tasks:

- Access layer properties for a layer.
- Evaluate properties from multiple layers.

Step 1: Access a feature service

In this step, you will use different content access methods to gain access to and examine a feature layer.

- a Open the ArcGIS Pro project from the previous exercise, if necessary.



Refer to the Exercise01 and Exercise02 notebooks for help.

- b Using your experience from the previous exercises, perform the following tasks:

- Create a new notebook named **Exercise03**.
- Use a code cell to import the GIS class.
- Insert a new code cell and add code to authenticate to the Training Services organization.
- Run the cells.

Next, you will explore how to access a feature layer and examine the properties associated with it.

- c In the Catalog pane, click the Portal tab.

- d Point to the RANCHO_BERNARDOTreeSurvey feature service and click the URL in the pop-up.

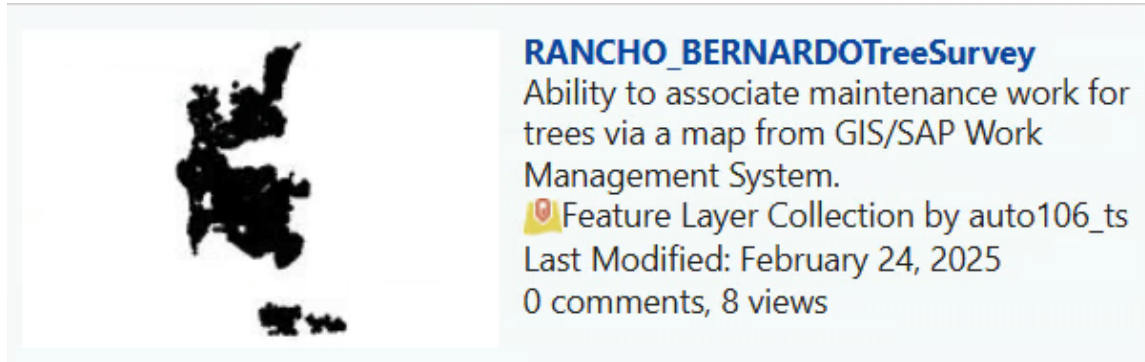


The item page for the survey displays in a web browser. If you are prompted to sign in, use the provided Training Services account.

- e On the right side of the item page, expand the Details section, if necessary, and locate the item ID.
- f Select and copy the ID.
- g Add code using the get method to access the feature service item.
- h On a new line in the same cell, type your variable to display the item.

```
trees = gis.content.get('0e899410cfe94ffa8ac1d8bcd497154b')
trees
```

- i** Run the cell.



The rich markup associated with the item should display as an output.

- j** In a web browser, go to the ArcGIS API for Python documentation: [Item class](#).
- k** Read the sentence about items with layers, and then answer the following questions.

1. Do you have access to an item or a feature collection?

2. Will the item contain a layers property?

Now that you know that this item can access layers contained within it, you will write code to examine different service properties associated with the feature layer.

- l** In a new code cell, type `trees`.



Include the period after `trees`.

- m** Press Tab to see the properties associated with the item.
- n** Use the down arrow to scroll through a few properties.
- o** Select the `layers` property and press Enter.
- p** Run the cell.

3. What type of data is returned from the `layers` property?

4. What API data type is contained within the list?

- q** In the next empty cell, write code to isolate the feature layer from the list in a variable called `treeLayer`.

You will examine the various properties to which you have access. These properties can be valuable with helping to manage fields and other metadata associated with the service.

Step 2: Examine feature layer properties

In this step, you will explore feature layer properties and test a range of capabilities.

- a** In the same cell, on a new line, enter the following code:

```
treeLayer.properties
```

- b** Run the cell.

The output contains more than just item information. The range of valuable properties can help you understand what this layer can do or access.

- c** Scroll through the properties output.
- d** Scroll to the bottom and answer the following questions.

5. Which capabilities are supported by the feature layer?

6. Could you edit this layer?

- e** In an empty cell, add code to test these properties to determine whether editing is enabled.

The following graphic shows an example of code to test the capability.

```
if "Editing" in treeLayer.properties['capabilities']:
    print("you can edit")
else:
    print("editing is not available")
```

- f Run the cell.

Understanding that this capability is enabled through code will be helpful when examining more than just a single feature layer. In addition to understanding what a layer can do, you will examine the fields associated with this layer.

- g Find the cell where you printed the layer properties.
- h Scroll through the output and find the fields property.

7. Which Python data type is returned if you request the fields property?

- i Scroll down to the last empty cell in the notebook.
- j In the empty cell, write code to print out the names of all the fields of the trees feature layer.

```
for fld in treeLayer.properties.fields:
    print(fld.name)
```

You have been asked to verify whether a status field is one of the fields in the tree service.

- k In the same cell with the field `for` loop, add code to test for the existence of the *Status* field.



You can accomplish this task with various Python techniques; use the technique that is most familiar to you. The following graphic shows how examples that use a loop or a list comprehension.


```
#for loop
for fld in treeLayer.properties.fields:
    if fld.name == "Status":
        print("Status field is there")
    else:
        print("Status field is not there")

# List comprehension
fldNames = [fld.name for fld in treeLayer.properties.fields]
"Status" in fldNames
```

In this step, you accessed the `layers` property object that shares valuable information about the service settings associated with the feature layer. This property object gives you insight into what is and is not enabled on the service. It also shows you information like `maxrecordcount` and the types of editing that may be allowed. Next, you will apply this capability to more feature layers.

Step 3: Verify field information for many layers

In this step, you will apply the same techniques to many tree survey layers within your organization.

- a Click the next empty code cell.
- b Set the cell type to Markdown.
- c Enter the following Markdown text:


```
### Search for Tree Survey Layers and Examine properties and fields
```
- d Run the Markdown cell.
- e In the blank cell, add code to perform the following tasks:
 - Search for all tree survey feature layers.
 - Loop on the returned layers.
 - Test whether the editing capability is enabled.
 - Verify the existence of the status field.
 - Report your findings.

Hint: Use the following resources:

- Solutions from Exercise 2 for searching
- Code from the previous steps to access properties
- Code samples from the ArcGIS API for Python API documentation: [Properties of FeatureLayer](#)

f Report your findings in the following table.

Layer name	Editing enabled	Status field

In this step, you combined resources to identify different services that will need updates.

- g** Save your Exercise03 notebook.
- h** Save your project and leave ArcGIS Pro open.

Lesson review

1. What does the capabilities property tell you about a feature layer?

2. You have been asked to examine the maximum record counts for feature layers within a group in your organization. What steps should your API code take to accomplish the task?

Answers to lesson 3 questions

Explore the features module (page 3-4)

1. What is returned with `FeatureLayer` properties?

An object with `FeatureLayer` properties

2. How can the properties be accessed?

From the dictionary or as individual field properties

3. What is returned from `get_unique_values` on the `FeatureLayer` class?

A Python list of unique values for a given attribute

4. Which API class can be accessed through the `container` property?

`FeatureLayerCollection` class

Exercise 3: Manage feature layer content (page 3-6)

1. Do you have access to an item or a feature collection?

Item

2. Will the item contain a `layers` property?

Yes. The item type is a feature layer collection.

3. What type of data is returned from the `layers` property?

Python list

4. What API data type is contained within the list?

Feature layer

5. Which capabilities are supported by the feature layer?

Query, Create, Delete, Update, Editing

6. Could you edit this layer?

Yes. The Editing capability is enabled for this layer.

Answers to lesson 3 questions (continued)

7. Which Python data type is returned if you request the fields property?

A Python list of dictionaries as a field

4

Strategies for updating feature layer properties

As the number of feature layers in your organization increases, understanding how to update layers in bulk becomes more important.

Topics covered

- Workflows for updating feature layer definitions

- Applying workflows to modify feature layer properties

Strategies for updating feature layer properties

Within an organization, feature layers often change from their original intent. For example, you may need to enable editing on a layer to support a project's needs, or you may need to add a new field to multiple layers. Typically, updates to layers can be managed easily through your organization's page in ArcGIS Online or ArcGIS Enterprise. However, there are times when the ArcGIS API for Python will offer the best approach.

You can change layers at either the feature service level or the feature layer level. The level where the change should take place depends on what needs to be updated.

When making modifications, it is important to assess where those modifications are applied in the Python API. Updating properties for feature layers is performed using a workflow to access definition methods.

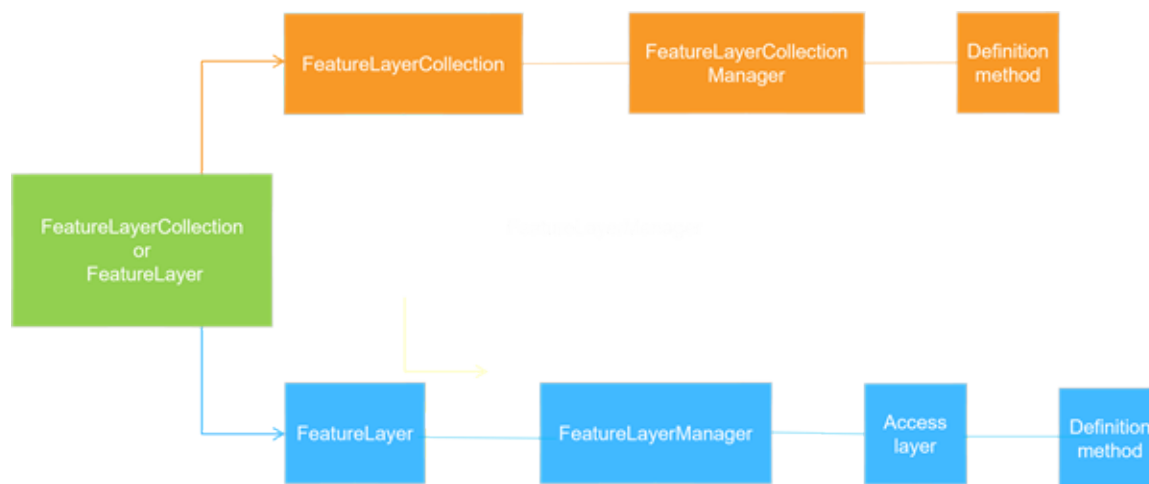


Figure 4.1. In this graphic, the orange workflow applies to updating properties for a feature service, while the blue workflow applies to updating properties for a feature layer.

Feature service changes

A feature service can be one or more layers and tables, and the service information can help define their uses. Changes to a feature service take place in the Python API through the `FeatureLayerCollection` class. The `FeatureLayerCollection` class enables the ability to read and make changes at the service level, enabling editor tracking, changing capabilities, and allowing the service to sync changes.

Strategies for updating feature layer properties (continued)

Feature layer changes

Feature layer changes are more granular, because changes are being made to a singular layer within the feature service. You can make the following changes at the feature layer level:

- Enable attachments
- Update the layer name
- Add a field
- Change the renderer

Manager helper objects

A manager helper object is used to provide administrative capabilities to a feature service and feature layer. The manager property provides access to methods that will update the definition of feature services and feature layers. Within the Python API documentation, definition changes that need to be made to the feature service are accessed through the `FeatureLayerCollectionManager` class. Changes that need to be made to an individual layer are accessed through the `FeatureLayerManager` class.

The following table describes the methods that both the `FeatureLayerCollectionManager` and `FeatureLayerManager` classes use to modify the definitions.

Method	Feature service	Feature layer
<code>add_to_definition</code>	<ul style="list-style-type: none"> • Create views • Create advanced views 	<ul style="list-style-type: none"> • Add fields to a layer • Create indexes on a layer
<code>delete_from_definition</code>	<ul style="list-style-type: none"> • Remove layers from service 	<ul style="list-style-type: none"> • Delete fields • Remove indexes
<code>update_definition</code>	<ul style="list-style-type: none"> • Turn on editor tracking • Enable sync 	<ul style="list-style-type: none"> • Rename a service layer • Enable attachments

Strategies for updating feature layer properties (continued)

Using the manager helper classes for a feature layer collection and feature layer requires the right authentication to use the methods provided. The Python API follows the same security settings that are used for the item in the organization. Access to the methods on the manager classes are available as follows:

- Item owner
- Organization administrator
- User role with ability to use methods to update definition



[ArcGIS API for Python documentation: Feature Layer properties](#)

[ArcGIS API for Python documentation: FeatureLayerManager](#)

[ArcGIS API for Python documentation: FeatureLayerCollectionManager](#)

Explore update capabilities in the FeatureLayer class

In this activity, you will use resources to explore how to update properties of feature layers. You will examine samples from the conceptual help and classes from the ArcGIS API for Python reference that you can use to update feature layers.

Instructions

- a** In a web browser, go to the ArcGIS API for Python documentation: [Update the feature service definition](#).
- b** Review the information on updating feature service properties and capabilities, and then answer the first two questions that follow.
- c** Go to the documentation for the [FeatureLayerManager](#) class.
- d** Review the information provided to answer the remaining questions.
- e** After you are finished, close the browser.

1. Which method can be used to modify feature layer properties?

2. Which data type is used to update the feature layer properties?

3. Where can the format for the first parameter of the `update_definition` method come from?

4. Do you need to use all layer properties for the `update_definition` method?

Modifying a feature layer definition

Feature layers are integral to ArcGIS Online and ArcGIS Enterprise deployment. As an organization grows, the number of feature layers will increase. The Python API provides a range of workflows that will allow for the automation of updates to services and layers.

Update feature layer properties

Now that you have learned about how to access the different manager classes, you will apply this knowledge to a feature layer collection and feature layer. Imagine that you are supporting an upcoming tree survey project within your organization, and some feature layers need new fields and updated properties. You will apply a workflow to add the necessary field and property changes.

In this exercise, you will perform the following tasks:

- Use the `FeatureLayerManager` class to add a field.
- Update feature layer properties.

Step 1: Add a status field to a single service

After examining the organization's tree survey feature layer, you identified a few layers that need to have a new Status field. In this step, you will update one service to include the new field.

- a Open the ArcGIS Pro project from the previous exercise, if necessary.
- b Open the Exercise04 notebook.
- c Add code to import the GIS class.
- d Add code to sign in to your organization.
- e Run the cells.

Before you add the fields to all the layers, you will test the update process on a single layer. In the previous exercise, you noticed that a field was missing from the Rancho Bernardo tree service. You will now add that field.

- f In an empty code cell, add the code to get the tree service and access the feature layer.



Use the Help resources or the Exercise03 notebook to help access the layer.

- g Run the cell to access the tree feature layer.
- h In an empty code cell, use the variable `flManager` and set it to the layers manager property.
- i Run the cell.

Changes to a feature layer using the Python API are completed through the `FeatureLayerManager` class. A feature layer manager is accessed by using the `manager` property of a feature layer.



The `manager` property is accessible only for layers that you can administer within the organization.

A `FeatureLayerManager` class can also be accessed using the `fromItem` method on the `FeatureLayerManager` class.

j In a web browser, review the following ArcGIS API for Python documentation links to understand how a field can be added to the feature layer:

- [FeatureLayerManager.add_to_definition](#)
- [Feature Layer properties](#)

1. **What type of data needs to be passed to the `add_to_definition` method to add a field?**

k In a new code cell, enter the following code to define the field to add:

```
adFLD = {'fields': [{'name': 'STATUS',
                      'type': 'esriFieldTypeString', 'length': 10,
                      'alias': 'Status', 'nullable': True, 'editable': True}]}
```

l On a new line in the same code cell, add the following code:

```
flManager.add_to_definition(adFLD)
```

m Run the cell.

If the field was added successfully, you will see a "success" JSON object in the cell output area.

In this step, you accessed the `FeatureLayerManager` class to add a missing field to a single feature layer. As you have been working with other layers, you noticed that they were also missing the STATUS field that needs to be part of the new schema.

Step 2: Add a status field to multiple services

In the previous step, you accessed a single layer and added a status field to a feature layer. In this step, you will apply the same workflow to multiple tree survey layers within your organization.

a Click the blank cell at the bottom of the Exercise04 notebook.

b Set the cell type to Markdown.

c Enter the following Markdown text:

```
### Search for Tree Survey Layers and Add the STATUS Field
```

d Run the Markdown cell.

This Markdown text provides context for what your code is about to do. It will help you and others who use the notebook to understand what is happening in the current process.

In the next step, you will apply the workflows from Step 1 to add a STATUS field to multiple feature layers within your organization.

e Write code to accomplish the following tasks:

- Search for all tree survey feature layers that your user owns.
- Loop on the returned layers.
- Verify the existence of the STATUS field.
- Add the field to the feature layer if it does not exist.

Hint: Use the following resources to help you write the code:

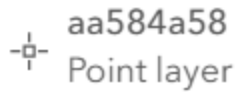
- Solutions from Exercise 3 for searching
- Code from the previous steps to access properties
- Code samples from the ArcGIS API for Python documentation: [Feature Layer properties](#)

In this step, you continued to combine resources and previously written code to modify feature layers that required a schema update.

Step 3: Change the layer name within the feature collection

When you access multiple feature layer collections in your organization, you might have noticed that the item name is correct, but some of the layer names have an automated value. To resolve this issue, you will revisit the code from the previous step. You will change the information required and use a different update method.


Layers



- a In the Exercise04 notebook, click the next empty cell.
- b Change the cell to a Markdown cell and enter the following text:

```
### Find and modify the layer names
```
- c Run the cell.

The tree survey layers have this automated layer name associated with them. You will reuse your code from Step 2 with changes to modify the layer names.

- d Select the code cell from Step 2.
- e Click the Create A Duplicate Of This Cell Below button  .
- f Click and hold to the left of the duplicated cell.
- g Drag the code cell below the Markdown cell.

Because this cell has most of the code that you need, only a few modifications are needed to modify the layer name.

- h Open a web browser to the following location: [FeatureLayer manager UpdateDefinition](#).

2. What is the recommendation for the `UpdateDefinition json_dict` parameter?

- i Delete the field loop and STATUS field check code from the new cell.
- j Comment out the `add_to_definition` code.

Your cell should look like the following graphic:

```

for treeItem in treesFL:

    treeLayer = treeItem.layers[0]

    flManager = treeLayer.manager

    #flManager.add_to_definition(adFLD)

```

- k** Add the following line of code within the loop:

```
print(treeLayer.properties)
```

- l** Run the cell.
- m** Examine the output from the code.

3. Which layer property needs to be added to the `json_dict` parameter?

After assessing the name, you have decided to concatenate the Item title and the static text `_SD`.

- n** Create a variable that is named `newLayerName`, and then use the Python technique of your choice to add the two values together.
- o** Comment out the print layer properties.
- p** Add a `print` function to print the new layer name for all layers.
- q** Run the cell.

The output should show your new proposed layer name.

- r** Comment out the `add_to_definition` line of code.
- s** Change the line to `update_definition`.
- t** Add the following dictionary to the `json_dict` parameter:


```
{ 'name':newLayerName }
```
- u** Run the cell.

- v Save your notebook and your ArcGIS Pro project.
- w In the Catalog pane, click the Portal tab.
- x Find the PACIFIC_BEACHTreeSurvey item.
- y Click the URL in the item pop-up.
- z In the opened browser, verify the change to the layer name.

In this exercise, you used different feature layer methods to manage fields and other metadata within your organization.

Lesson review

1. What is accomplished by the following code?

```
from arcgis.features import FeatureLayerCollection
item = gis.content.get('4dbf71d136264919b028785e49617c8a')
itemFC = FeatureLayerCollection.fromitem(item)
itemFC.manager.update_definition
({"capabilities": "Create, Query, Update"})
```

2. What is accomplished by the following code?

```
item = gis.content.get('af35bbd33816488b8444a2c63b752ee3')
fld = {"fields": [{"name": "status"}]}
item.layers[0].manager.delete_from_definition(fld)
```

Answers to lesson 4 questions

Explore update capabilities in the FeatureLayer class (page 4-5)

1. Which method can be used to modify feature layer properties?

`update_definition`

2. Which data type is used to update the feature layer properties?

Dictionary of properties

3. Where can the format for the first parameter of the `update_definition` method come from?

From the feature layer properties property

4. Do you need to use all layer properties for the `update_definition` method?

No—only the properties to be updated

Exercise 4: Update feature layer properties (page 4-7)

1. What type of data needs to be passed to the `add_to_definition` method to add a field?

JSON dictionary

2. What is the recommendation for the `UpdateDefinition json_dict` parameter?

Use the format of the layer properties.

3. Which layer property needs to be added to the `json_dict` parameter?

Name

5

Strategies for editing feature layers

As you work with feature layers in your organization, you will find the need to make changes to features that require the use of the ArcGIS API for Python. Those changes might apply to a single layer or multiple layers in an organization. This lesson will focus on creating, updating, and deleting features associated with a layer, and you will apply workflows for editing features within a feature layer.

Topics covered

- Workflows to update features in feature layers

- Applying workflows to modify features in feature layers

Strategies for editing feature layers

As your organization adopts new applications and workflows to assist your goals with GIS, you might find a need to automate feature editing. Whether you are creating new features obtained from different sources or removing features that no longer need to be maintained, the ArcGIS API for Python has classes and methods available to support these workflows.

If you are familiar with editing in ArcGIS Online or ArcGIS Enterprise, you know that editing starts with a feature layer. What type of editing can be performed depends on how the service is being administered. The type of editing that can be performed on a feature layer is determined by the capabilities set by the administrator.

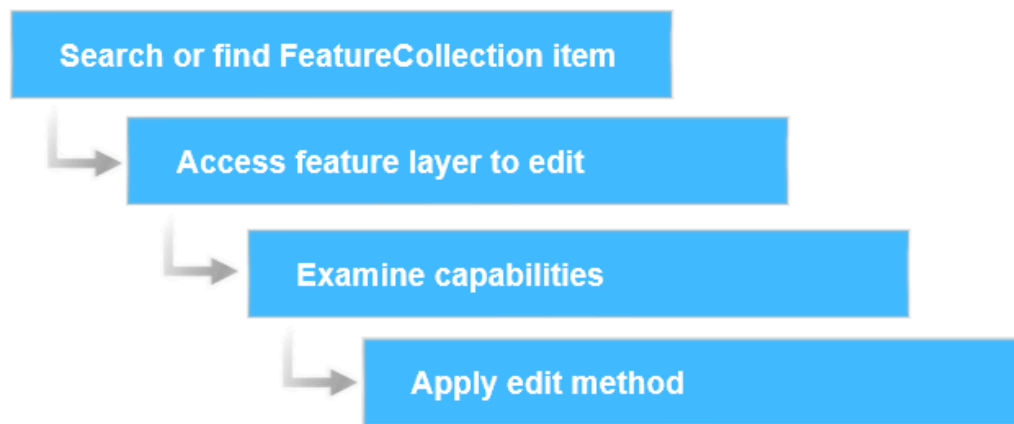


Figure 5.1. This graphic illustrates the pseudo code that you might use to edit features in your code.

Capabilities

When a feature service is created, the publisher creates the service with a purpose in mind. When it comes to supporting editing, a decision needs to be made about what kind of edits the feature layer will support. The author of the layer will enable editing and determine the types of edits the feature layer will support. The following table outlines the different edits that a feature layer can support.

Strategies for editing feature layers (continued)

Edit type	Purpose
Add	Creates new features
Delete	Removes features from the service
Update	Changes attributes, or changes both attributes and geometry

When writing code that will edit a feature layer within the Python API, it can be important to verify the types of edits allowed to the service. If your code attempts to make an edit that is not allowed by the service, an error will occur. You need to be able to understand if the error was in your code or in the service capabilities.

Editing methods

A feature layer class in the Python API supports a variety of editing methods. It is important to match the editing method to what you are trying to accomplish with your edits.

Edit method	Purpose
<code>append</code>	<ul style="list-style-type: none"> • Appends data from another item into a target service • Supports additional parameters for managing updates and inserts • Good for large data updates
<code>calculate</code>	<ul style="list-style-type: none"> • Attribute updates only. • Layer must have <code>supportsCalculate</code> enabled
<code>delete_features</code>	<ul style="list-style-type: none"> • Deletes features when the service supports delete • Using a <code>'1=1'</code> expression in the <code>where</code> parameter will delete all features
<code>edit_features</code>	<ul style="list-style-type: none"> • Supports the ability to create update and delete features • <code>FeatureSet</code> input is required to identify features to be updated

Strategies for editing feature layers (continued)

Adding features



A common process with the Python API is to create new features and add them to existing feature layers. The editing method that supports this operation is the `edit_features` method, available from a feature layer class in the Python API.

The first step in that process is to engineer the data into a feature format. The feature format is a Python dictionary of attributes and geometry. The Python language supports a variety of ways to arrange the data in the needed format. You will also find the Python API has classes and methods that will help to arrange the data in the correct format.

When the individual records are created, they need to be stored within a larger list. This list of features is known as a feature set, and it represents the features to add to the service. The `adds` parameter requires this input.

The new features are applied to the service using the `adds` parameter of the `edit_features` method.

After the edits are applied to the service, it is important to verify with code or visually that the features are available in the service.

Updating features



As a feature layer is used more, updates will be needed to update geometries, attributes, or both. This process is an active part of data maintenance.

When updating features through the `edit_features` method, the first step is important. If you

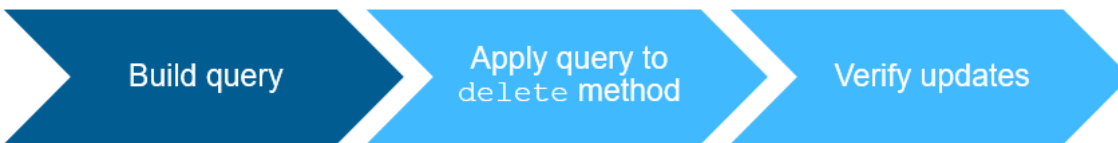
Strategies for editing feature layers (continued)

are not updating all features, the features that will be updated need to be isolated or identified. The `query` method from the feature layer class can help with this process. The `query` method can return a `FeatureSet` of features. Another option is to use the Spatially Enabled DataFrame (SEDF) to select the features to be updated and exported to a `FeatureSet`.

After the features that need the geometry or attribute changes have been identified, the changes need to be applied to those features. Identify the feature attribute or geometry that needs to be updated and apply it to the data.

The updates to the features are applied to the service using the `updates` parameter of the `edit_features` method.

Deleting features



Removing features from a service helps keep a service up to date. An important consideration when removing features is making sure that they need to be removed. A delete operation can have bigger impacts on other parts of your organizational workflows. A delete operation can be performed with `edit_features` or `delete_features`. The `delete_features` method supports a `where` parameter that provides more flexibility for the edit operation.

When using the `delete_features` method, a query will be applied to the data to find the features to remove from the service. The typical ArcGIS SQL queries can be applied for the `where` parameter.

The query is applied to the data through the `delete_features` method, with the query as the parameter.

A message is returned from the feature layer indicating whether the delete operation was successful. An additional visual verification can be done to ensure that the features were removed.

Explore edit capabilities in the FeatureLayer class

In this activity, you will use resources to explore how to update properties of feature layers. You will examine samples from the conceptual help and classes from the Python API reference that you can use to update feature layers.

Instructions

- a** Open a web browser and go to the ArcGIS API for Python documentation: [edit features](#).
- b** After you are finished, close the web browser.

1. If an edit operation is over 250 features, which method should be used in place of `edit_features`?

2. If `edit_features` is used to update data, which data type needs to be provided to the `updates` parameter?

Feature Service data structures

Different data structures are used in creating, updating, and deleting features. Various Help resources can help you to understand both how data can be shaped for editing with feature layers and how you can edit features for a feature layer.

Edit feature layers

You are continuing to support the tree survey project within your organization, and some feature layers in the organization need new features, while others require updates and feature removals. You will apply various editing workflows to make the necessary feature changes.

In this exercise, you will perform the following tasks:

- Add new features with the `FeatureLayer` class.
- Modify features from a feature layer.
- Remove features from a feature layer.

Step 1: Engineer input data

An evaluation of the downtown tree service was completed and revealed missing features around the downtown ballpark. Someone in your organization emailed you a CSV file that contains the new features to add to the downtown tree service. In this step, you will access the feature layer and engineer the data.

- a Open the ArcGIS Pro project from the previous exercise, if necessary.
- b From the Catalog pane, open the Exercise05 notebook.
- c Run the first two code cells in the notebook.
- d Find the code cell that defines the `downTownTreeItem` variable.
- e In the Catalog pane, on the Portal tab, point to the DOWNTOWNTreeSurvey service.
- f Click the item URL in the pop-up.
- g On the right side of the page, expand the Details section and copy the item ID.
- h Return to the Exercise05 notebook and replace the `<ItemID>` text with your copied item ID.
- i Run the cell.

1. Which property in the feature layer capabilities can help you determine whether editing is enabled for a feature layer?

- j In a new code cell, access the capabilities from the layer properties.
- k After writing the code, run the code cell.



Use the solution code or your notebook from Exercise 3 or 4 for help.

2. Does the DOWNTOWNTreeSurvey service support the Editing capability?

To add new features, you will access a CSV file that contains the BallPark Trees features that you

will add to the service. You will use pandas and the Spatially Enabled DataFrame. The Spatially Enabled DataFrame has methods to easily transform data into different formats needed for different operations.

l In a new code cell, enter the following code:

```
newTrees = r"C:\EsriTraining\ARPY\Ballparktrees.csv"
```

m In the same code cell, on a new line, enter the following code:

```
df = pd.read_csv(newTrees)
```

n In the same code cell, on a new line, enter the following code:

```
newTreesSedf = pd.DataFrame.spatial.from_xy  
(df, x_column="X", y_column="Y", sr=2230)
```

o In the same code cell, on a new line, enter the following code:

```
newTreesSedf.head()
```

p Run the current code cell.

The first five records within the Spatially Enabled DataFrame are shown in the cell output. If you see something different or received an error, verify your code against the preceding actions.



The 2230 listed in the parameters is the WKID for the NAD 1983 StatePlane coordinate system, which used for the feature layer. This parameter helps the Spatially Enabled DataFrame to work correctly with the geometries associated within the CSV file.

When you add features, it is important to verify that the fields match between the feature layer and the features to be added. Next, you will add code that will help compare the fields between the layer and the Spatially Enabled DataFrame.

q In a new code cell, enter the following code:

```
lyrFldNames = [fld.name for fld in downTownTreeLayer.properties.fields]
```

r In the same code cell, on a new line, enter the following code:

```
sedfCols = list(sedf.columns)
```


- s In the same code cell, on a new line, enter the following code:

```
notInSEDF = set(sedfCols) - set(lyrFldNames)
```

- t In the same code cell, on a new line, enter the following code:

```
notInSEDF
```

- u Verify that your code matches the following graphic.

```
lyrFldNames = [fld.name for fld in downTownTreeLayer.properties.fields]

sedfCols = list(newTreesSedf.columns)

notInSEDF = set(sedfCols) - set(lyrFldNames)

notInSEDF
```

- v Run the current code cell.
- w Compare your results to the following graphic.

```
Out[13]: {'ID', 'ID2', 'SHAPE', 'ID3', 'X', 'Y'}
```

When the code runs, it compares the fields from the feature layer to the fields from the CSV file. Using the Python data types, you can compare the differences between the two lists. The outcome of the difference is a list of the fields that should be dropped from the Spatially Enabled DataFrame.

Step 2: Add features

In the previous step, you engineered the data by creating a Spatially Enabled DataFrame to compare the fields in the file to the service. In this step, you will create the new tree features.

- a In a new code cell, enter the following code:

```
newTreesSedf.drop(['ID', 'ID2', 'ID3', 'X', 'Y'], axis=1, inplace=True)
```

- b In the same code cell, on a new line, enter the following code:

```
newTreesSedf.info()
```

- c Run the current code cell.

This code will drop those columns from the DataFrame. The `info` method helps verify that the columns are no longer part of the DataFrame. At this point, the new data is ready to be converted and added as new features to the service.

- d In a new code cell, enter the following code:

```
newTrees = newTreesSedf.spatial.to_featureset()
```

- e In the same code cell, on a new line, enter the following code:

```
downTownTreeLayer.edit_features(adds=newTrees)
```

- f Run the current code cell.

The output portion of the code cell shows the results of the editing operation. If any errors took place during the creation process, they would be posted in the cell output.

- g Save your notebook.
- h In the Catalog pane, on the Project tab, expand Maps.
- i Open the DownTown Trees map.
- j In the Catalog pane, click the Portal tab.
- k Locate the DOWNTOWNTreesSurvey feature service.
- l Drag DOWNTOWNTreeSurvey into the map and wait for the service to finish drawing.
- m On the ribbon, from the Map tab, click Bookmarks and choose the Ballpark Area bookmark.



If your downtown tree service does not reflect these changes, look back through the outputs of the previous cells for typos and error messages.

In this step, you used a feature layer update method that helped keep the layer up to date with a small number of new features. New trees were added along Tony Gwynn Drive, 10th Avenue, and Park Blvd, as well as around the park in the park area. Next, you will explore how to remove features to update a feature layer.

Step 3: Update features

An analyst in your organization went to the ballpark to verify the new additions to the downtown

trees service. Upon inspection, it was noticed that some feature attributes were missing. In this step, you will identify the features and update their attributes.

- a In a web browser, go to the ArcGIS API for Python documentation: [arcgis.features module.query](#).
- b Examine the `where` and `out_fields` parameters, as well as the type of data returned from the query.

3. How are the fields passed to the `out_fields` parameter?

4. The species value for new trees is Null. Which SQL statement could query those features?

5. Which data type is returned by the query method?

- c Return to the Exercise05 notebook.
- d In a new code cell, enter the following line of code:


```
treesToCalc = downTownTreeLayer.query
(where="SPECIES=Null",out_fields=["OBJECTID","SPECIES"])
```
- e In the same code cell, on a new line, enter the following line of code:


```
treesToCalc
```
- f Run the current code cell.

In the cell output area, the query should return more than 50 features that should have a SPECIES attribute that is Null.

- g In a web browser, go to the ArcGIS API for Python documentation: [FeatureLayer.edit_features](#).
- h Examine the information about the `updates` parameter.

6. Which data type needs to be supplied to the `updates` parameter?

The query method returned all the features that need to change; however, the SPECIES attribute is currently Null.

i Minimize the browser and return to the Exercise05 notebook in ArcGIS Pro.

j In a new code cell, enter the following line of code:

```
for tree in treesToCalc.features:
```

k In the same code cell, on a new line, enter and indent the following line of code:

```
    tree.attributes['SPECIES'] = 'PALM'
```

l Ensure that your code looks like the following graphic.

```
for tree in treesToCalc.features:
    tree.attributes['SPECIES'] = "PALM"
```

m Run the current code cell.

The changes have been made to the features in the feature set; however, changes have not been applied back to the service.

n In a new code cell, enter the following line of code:

```
downTownTreeLayer.edit_features(updates=treesToCalc.features)
```

o Run the current code cell.

The output portion of the code cell shows the results of the editing operation. If any errors took place during the creation process, they would be posted in the cell output. To test if there are any features with SPECIES = Null, you can re-run the code cell with the query method and verify the cell output.

p Re-run the code with the query looking for Species = Null.

q Examine the output of the cell.

If the update operation was successful, the result of the query should be 0 features.

Step 4: Delete features

After collecting some new data in the Northern County area of San Diego, it has been determined that new updates need to be made to the Rancho Bernardo area. Looking at the data, there are different tree species in the survey that need to be re-assessed. In this step, you will access the feature layer and remove features with the Python API.

- a At the bottom of the Exercise05 notebook, create a new cell.
- b Change the cell to a Markdown cell.
- c Enter the following Markdown text:

```
## Remove Trees from a service
```
- d Run the cell.

If you recall, the first step in the editing process is to access the feature layer where the edits will take place.

- e Use the previous exercise solutions and ArcGIS API for Python documentation to find or access the RANCHO_BERNARDOTreeSurvey feature layer.
- f Use the `RBTreeLayer` variable to reference the Trees feature layer.
- g Use the layer capabilities to determine whether editing is allowed on the feature layer.
- h Enter your code in a single code cell.
- i Run the code cell when complete.

7. Examining the capabilities, what indicates that the Trees feature layer has the ability to delete features?

In the following actions, you will use the `delete_features` method to remove the required trees from the feature layer. You got notes from an arborist that you need to reassess the following tree types: BRADFORD PEAR, RED IRONBARK, and JACARANDA.

- j In a web browser, go to the ArcGIS API for Python documentation:
[`FeatureLayer.delete_features`](#).

- k** Examine the parameters for running the method.

8. Which parameter would help to delete multiple tree species?

9. What would the SQL statement look like to delete the three types of trees?

- l** In a new code cell, enter the following code:

```
delSQL = "SPECIES = 'BRADFORD PEAR' OR SPECIES = 'RED IRONBARK' OR  
SPECIES = 'JACARANDA'"
```

- m** In the same code cell, on a new line, enter the following code:

```
RBTreeLayer.delete_features(delSQL)
```

The output portion of the code cell shows the results of the editing operation. If any errors took place during the creation process, they would be posted in the cell output.

- n** In a new code cell, enter the following code:

```
RBTreeLayer.query(where=delSQL)
```

- o** Run the current code cell.



If features were returned, check your code from the previous step and make the correct changes, then run the cells to remove the tree features.

If the previous edit operation was successful, zero features will be returned from the query.

- p** Save the ArcGIS Pro project.

In this exercise, you used different editing techniques to manage features in a feature layer.

Alternative strategies to edit feature layers

In the following scenarios, you will think about and use these resources to help you determine best approaches for editing features within feature layers:

- ArcGIS API for Python documentation: [edit_features](#)
- ArcGIS API for Python documentation: [append](#)

Scenario 1: Removing features from a feature layer

After verifying its use in your organization, you have to remove all the features from a feature layer.

1. What steps would you take to remove the features from the feature layer?

Scenario 2: Adding a large number of new features

You are working with a colleague in your organization who has a large number of sampling point data stored in a shapefile. It has been determined that the data should be added to an existing sampling point service.

2. What editing processes can be used to add the data to the service?

Lesson review

1. After running the `edit_features` method in a notebook, no results are shown in the output. You decide to check the capabilities and see the following result.

```
"tileMaxRecordCount": 8000,  
"maxRecordCountFactor":1,  
"capabilities": "Create,Delete,Query,Update,Editing"
```

What additional editing setting can you check to understand why the edit was unsuccessful?

2. Which edit method works on only attributes?
-

Answers to lesson 5 questions

Explore edit capabilities in the `FeatureLayer` class (page 5-6)

1. If an edit operation is over 250 features, which method should be used in place of `edit_features`?

The `append` method should be used over `edit_features` to improve performance and ensure service stability.

2. If `edit_features` is used to update data, which data type needs to be provided to the `updates` parameter?

Optional `FeatureSet/List`; the array of features to be updated

Exercise 5: Edit feature layers (page 5-8)

1. Which property in the feature layer capabilities can help you determine whether editing is enabled for a feature layer?

Editing

2. Does the `DOWNTOWNTreeSurvey` service support the Editing capability?

Yes.

3. How are the fields passed to the `out_fields` parameter?

Python list of field names

4. The species value for new trees is Null. Which SQL statement could query those features?

"SPECIES=Null"

5. Which data type is returned by the query method?

`FeatureSet`

6. Which data type needs to be supplied to the `updates` parameter?

`FeatureSet`

Answers to lesson 5 questions (continued)

7. Examining the capabilities, what indicates that the Trees feature layer has the ability to delete features?

Delete and Editing are visible in the capabilities.

8. Which parameter would help to delete multiple tree species?

A where clause for the query filter

9. What would the SQL statement look like to delete the three types of trees?

```
"SPECIES = 'BRADFORD PEAR' OR SPECIES = 'RED IRONBARK' OR SPECIES =  
'JACARANDA' "
```

Alternative strategies to edit feature layers (page 5-17)

Scenario 1: Removing features from a feature layer

1. What steps would you take to remove the features from the feature layer?

1. Access the feature layer collection.
2. Reference the individual layer with the features to remove.
3. Verify editing and delete is supported.
4. Use the delete_features with a where='1=1' parameter.

Scenario 2: Adding a large number of new features

2. What editing processes can be used to add the data to the service?

The append process would be the most appropriate method of adding the features to the service.

If the number is below 250, then edit_features add could be available.

Appendix A

Data license agreement

ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE, INC. ("ESRI"), IS WILLING TO LICENSE THE ENCLOSED ELECTRONIC VERSION OF THE TRAINING MATERIALS TO THE STUDENT ("YOU") ONLY UPON THE CONDITION THAT YOU ACCEPT ALL TERMS AND CONDITIONS CONTAINED IN THIS ESRI DATA LICENSE AGREEMENT ("AGREEMENT"). PLEASE READ THE TERMS AND CONDITIONS CAREFULLY. BY CLICKING, "I ACCEPT", YOU ARE INDICATING YOUR ACCEPTANCE OF THE ESRI DATA LICENSE AGREEMENT. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS AS STATED, ESRI IS UNWILLING TO LICENSE THE TRAINING MATERIALS TO YOU.

Training Materials Reservation of Ownership. This Agreement gives You certain limited rights to use electronic and tangible versions of the digital or printed content required to complete a course, which may include, but are not limited to, workbooks, data, concepts, exercises, and exams ("Training Materials"). Esri and its licensor(s) retain exclusive rights, title, and ownership to the copy of Training Materials, software, data, and documentation licensed under this Agreement. Training Materials are protected by United States copyright laws and applicable international copyright treaties and/or conventions. All rights not specifically granted in this Agreement are reserved to Esri and its licensor(s).

Grant of License. Esri grants to You a personal, nonexclusive, nontransferable license to use Training Materials for Your own training purposes. You may run and install one (1) copy of Training Materials and reproduce one (1) copy of Training Materials. You may make one (1) additional copy of the original Training Materials for archive purposes only, unless Esri grants in writing the right to make additional copies.

Training Materials are intended solely for the use of the training of the individual who registered and attended a specific training course. You may not (i) separate the component parts of the Training Materials for use on multiple systems or in the cloud, use in conjunction with any other software package, and/or merge and compile into a separate database(s) or documents for other analytical uses; (ii) make any attempt to circumvent the technological measure(s) (e.g., software or hardware key) that effectively controls access to Training Materials; (iii) remove or obscure any copyright, trademark, and/or proprietary rights notices of Esri or its licensor(s); or (iv) use audio and/or video recording equipment during a training course.

Appendix A

Data license agreement (continued)

Term. The license granted by this Agreement will commence upon Your receipt of the Training Materials and continue until such time that (1) You elect to discontinue use of the Training Materials or (2) Esri terminates this Agreement for Your material breach of this Agreement. This Agreement will be terminated automatically without notice if You fail to comply with any provision of this Agreement. Upon termination of this Agreement in either instance, You will return to Esri or destroy all copies of the Training Materials, including any whole or partial copies in any form, and deliver evidence of such destruction to Esri, and which evidence will be in a form acceptable to Esri in its sole discretion. The parties hereby agree that all provisions that operate to protect the rights of Esri and its licensor(s) will remain in force should breach occur.

Limited Warranty. Esri warrants that the media on which Training Materials is provided will be free from defects in materials and workmanship under normal use and service for a period of ninety (90) days from the date of receipt.

Disclaimer of Warranties. EXCEPT FOR THE LIMITED WARRANTY SET FORTH ABOVE, THE TRAINING AND TRAINING MATERIALS CONTAINED THEREIN ARE PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. ESRI DOES NOT WARRANT THAT THE TRAINING OR TRAINING MATERIALS WILL MEET YOUR NEEDS OR EXPECTATIONS; THAT THE USE OF TRAINING MATERIALS WILL BE UNINTERRUPTED; OR THAT ALL NONCONFORMITIES, DEFECTS, OR ERRORS CAN OR WILL BE CORRECTED. THE TRAINING DATABASE HAS BEEN OBTAINED FROM SOURCES BELIEVED TO BE RELIABLE, BUT ITS ACCURACY AND COMPLETENESS, AND THE OPINIONS BASED THEREON, ARE NOT GUARANTEED. THE TRAINING DATABASE MAY CONTAIN SOME NONCONFORMITIES, DEFECTS, ERRORS, AND/OR OMISSIONS. ESRI AND ITS LICENSOR(S) DO NOT WARRANT THAT THE TRAINING DATABASE WILL MEET YOUR NEEDS OR EXPECTATIONS, THAT THE USE OF THE TRAINING DATABASE WILL BE UNINTERRUPTED, OR THAT ALL NONCONFORMITIES CAN OR WILL BE CORRECTED. ESRI AND ITS LICENSOR(S) ARE NOT INVITING RELIANCE ON THIS TRAINING DATABASE, AND YOU SHOULD ALWAYS VERIFY ACTUAL DATA, SUCH AS MAP, SPATIAL, RASTER, OR TABULAR INFORMATION. THE DATA CONTAINED IN THIS PACKAGE IS SUBJECT TO CHANGE WITHOUT NOTICE. IN ADDITION TO AND WITHOUT LIMITING THE PRECEDING PARAGRAPH, ESRI DOES NOT WARRANT IN ANY WAY TRAINING DATA. TRAINING DATA MAY

Appendix A

Data license agreement (continued)

NOT BE FREE OF NONCONFORMITIES, DEFECTS, ERRORS, OR OMISSIONS; BE AVAILABLE WITHOUT INTERRUPTION; BE CORRECTED IF ERRORS ARE DISCOVERED; OR MEET YOUR NEEDS OR EXPECTATIONS. YOU SHOULD NOT RELY ON ANY TRAINING DATA UNLESS YOU HAVE VERIFIED TRAINING DATA AGAINST ACTUAL DATA FROM DOCUMENTS OF RECORD, FIELD MEASUREMENT, OR OBSERVATION.

Exclusive Remedy. Your exclusive remedy and Esri's entire liability for breach of the limited warranties set forth above will be limited, at Esri's sole discretion, to (i) replacement of any defective Training Materials; (ii) repair, correction, or a workaround for Training Materials; or (iii) return of the fees paid by You for Training Material that do not meet Esri's limited warranty, provided that You uninstall, remove, and destroy all copies of the Training Materials and execute and deliver evidence of such actions to Esri.

IN NO EVENT WILL ESRI BE LIABLE TO YOU FOR COSTS OF PROCUREMENT OF SUBSTITUTE GOODS OR TRAINING; LOST PROFITS; LOST SALES; BUSINESS EXPENDITURES; INVESTMENTS; BUSINESS COMMITMENTS; LOSS OF ANY GOODWILL; OR ANY INDIRECT, SPECIAL, EXEMPLARY, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING OUT OF OR RELATED TO THIS AGREEMENT, HOWEVER CAUSED OR UNDER ANY THEORY OF LIABILITY, EVEN IF ESRI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. ESRI'S TOTAL CUMULATIVE LIABILITY HEREUNDER, FROM ALL CAUSES OF ACTION OF ANY KIND, WILL IN NO EVENT EXCEED THE AMOUNT ACTUALLY PAID BY YOU FOR THE PORTION OF THE TRAINING UNDER THIS AGREEMENT. THESE LIMITATIONS WILL APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY.

Export Regulation. You must comply with all applicable laws and regulations of the United States including, without limitation, its export control laws. You expressly acknowledge and agree not to export, reexport, transfer, or release Esri-provided Training Materials, in whole or in part, to (i) any US embargoed country (including to a resident of any US embargoed country); (ii) any person or entity on the US Treasury Department Specially Designated Nationals List; (iii) any person or entity on the US Commerce Department Lists of Parties of Concern; or (iv) any person or entity where such export, reexport, or provision violates any US export control laws or regulations including, but not limited to, the terms of any export license or licensing provision and any amendments and supplemental additions to US export laws.

Appendix A

Data license agreement (continued)

Governing Law. This Agreement is governed by and construed in accordance with the laws of the state in which training is being held or, in the case of training provided over the Internet, the laws of the State of California, without reference to its conflict of laws principles.

Appendix B

Answers to lesson review questions

Answers to lesson 1 review questions

1. What is accomplished by the following code?

```
from arcgis.gis import GIS
enterprise = GIS("https://ebase.ad.local/portal",
profile='MyEnterLogin', verify_cert=False)
```

The code will sign in to ArcGIS Enterprise using a stored profile. It will ignore any certificate issues that may be encountered.

2. What is accomplished by the following code?

```
from arcgis.gis import GIS
gis = GIS("https://www.arcgis.com", profile='ProductionOrg',
set_active=True)
```

The code will sign in to ArcGIS Online using a stored profile. The `set_active` parameter defines that this will be the default organization used, if more than one organization is signed in with the script.

Answers to lesson 2 review questions

1. What is returned by the following code?

```
gis.content.search(query="owner:" + gis.users.me.username,
item_type="Feature Layer",
max_items=15)
```

A Python list of 15 feature layers in My Contents with the AUTO tag associated

Appendix B

Answers to lesson review questions (continued)

2. You need to identify feature layers that fall within a new project boundary in your organization. Which method will help you to identify those feature layers?
b. `advanced_search`

Answers to lesson 3 review questions

1. What does the `capabilities` property tell you about a feature layer?
It indicates the enabled functions that a layer supports.
2. You have been asked to examine the maximum record counts for feature layers within a group in your organization. What steps should your API code take to accomplish the task?
 1. Use a search with the group filter and the item type filter.
 2. Write a loop to visit each item.
 3. Use the `layers` property to access each layer.
 4. Examine the `maxRecordCount` property from the layer properties.

Answers to lesson 4 review questions

1. What is accomplished by the following code?

```
from arcgis.features import FeatureLayerCollection
item = gis.content.get('4dbf71d136264919b028785e49617c8a')
itemFC = FeatureLayerCollection.fromitem(item)
itemFC.manager.update_definition
({"capabilities": "Create, Query, Update"})
```

Enabling Create and Update editing capabilities on a feature service

Appendix B

Answers to lesson review questions (continued)

2. What is accomplished by the following code?

```
item = gis.content.get('af35bbd33816488b8444a2c63b752ee3')
fld = {"fields":[{"name":"status"}]}
item.layers[0].manager.delete_from_definition(fld)
```

The code is deleting the status field from the first layer in a feature service.

Answers to lesson 5 review questions

1. After running the `edit_features` method in a notebook, no results are shown in the output. You decide to check the capabilities and see the following result.

```
"tileMaxRecordCount": 8000,
"maxRecordCountFactor":1,
"capabilities": "Create,Delete,Query,Update,Editing"
```

What additional editing setting can you check to understand why the edit was unsuccessful?

Explore the settings on the feature layer item page. Verify what type of updates are supported.

2. Which edit method works on only attributes?

The calculate method calculates only attributes, not geometries.

Appendix C

Additional resources

Lesson 1	Resources
Python libraries	<ul style="list-style-type: none">• ArcGIS API for Python documentation: <code>GeoAccessor.from_featureclass</code>• ArcGIS API for Python documentation: <code>FeatureSet.from_arcpy</code>• ArcGIS API for Python documentation: <code>GeoAccessor.from_table</code>• ArcGIS API for Python documentation: <code>GeoAccessor.select</code>

Lesson 2	Resources
Content management workflows	<ul style="list-style-type: none">• ArcGIS API for Python documentation: property content• ArcGIS API for Python documentation: <code>ContentManager</code>• ArcGIS API for Python documentation: Accessing and creating content• ArcGIS API for Python documentation: Managing your content
Using the <code>ContentManager</code> class	<ul style="list-style-type: none">• ArcGIS API for Python documentation: Accessing and creating content• ArcGIS API for Python documentation: <code>ContentManager</code> class

Appendix C

Additional resources (continued)

Lesson 4	Resources
Strategies for updating feature layer properties	<ul style="list-style-type: none">• ArcGIS API for Python documentation: Feature Layer properties• ArcGIS API for Python documentation: FeatureLayerManager• ArcGIS API for Python documentation: FeatureLayerCollectionManager

Contact Esri Training Services

T 909 793 2853 x1585

F 909 793 5953

esri.com/training

380 New York Street

Redlands, California 92373-8100 USA



esri

THE
SCIENCE
OF
WHERE®