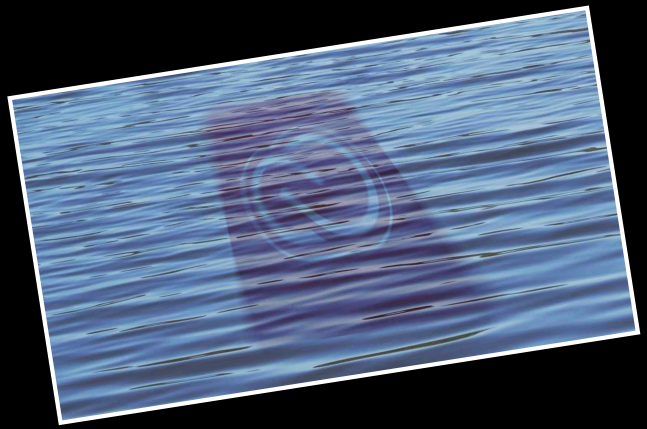
The Adobe User Sync Tool

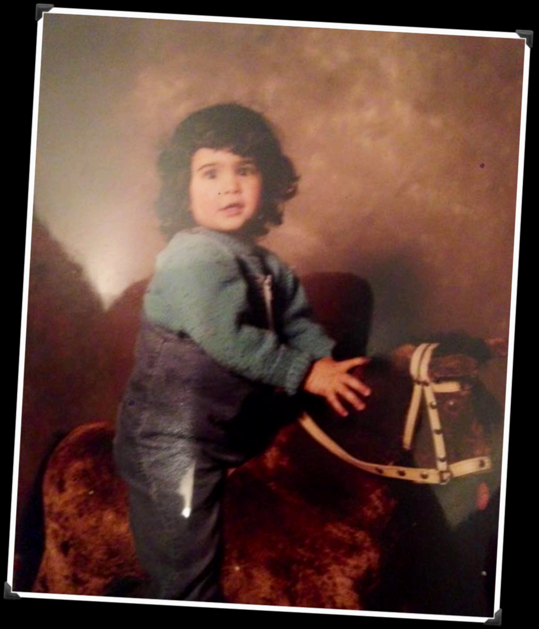
I’ve got that syncing feeling…



Who am I?

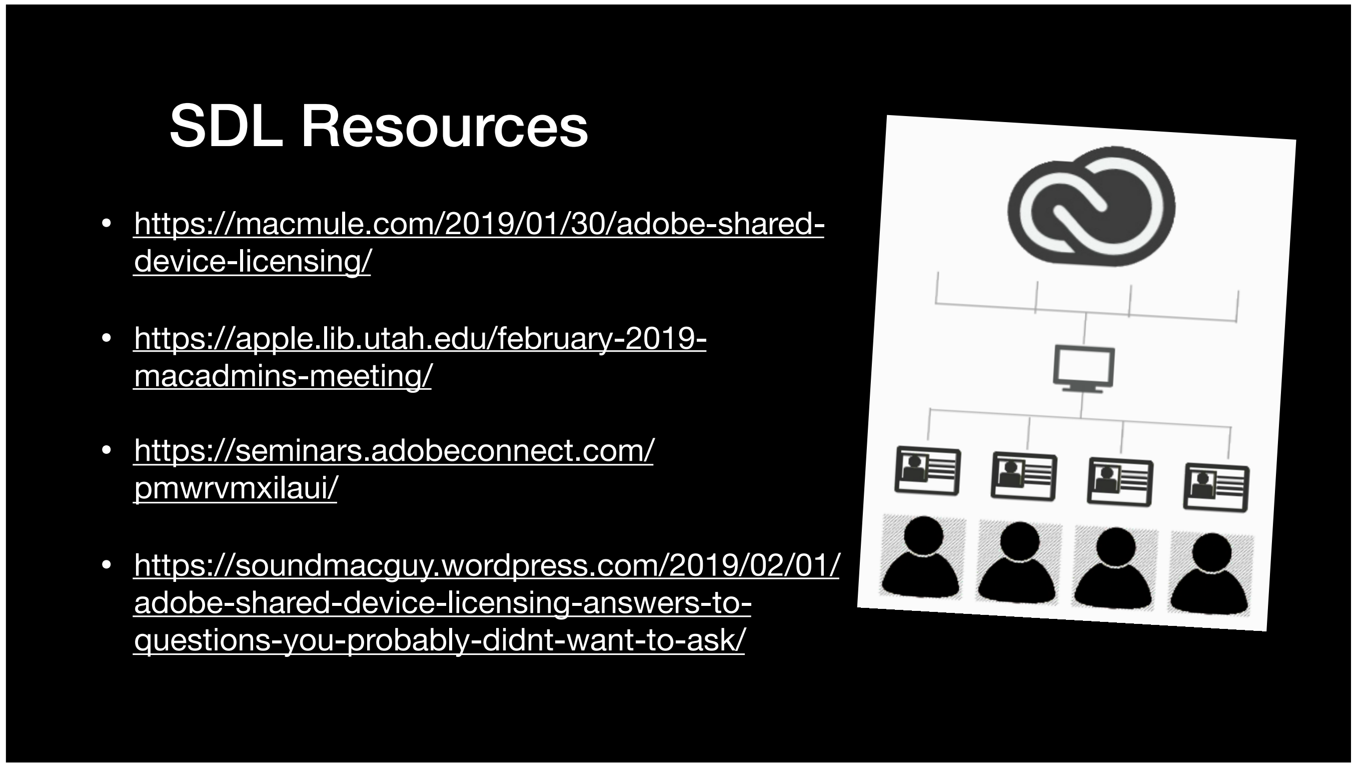
@neilmartin83

soundmacguy.wordpress.com



Before we begin…

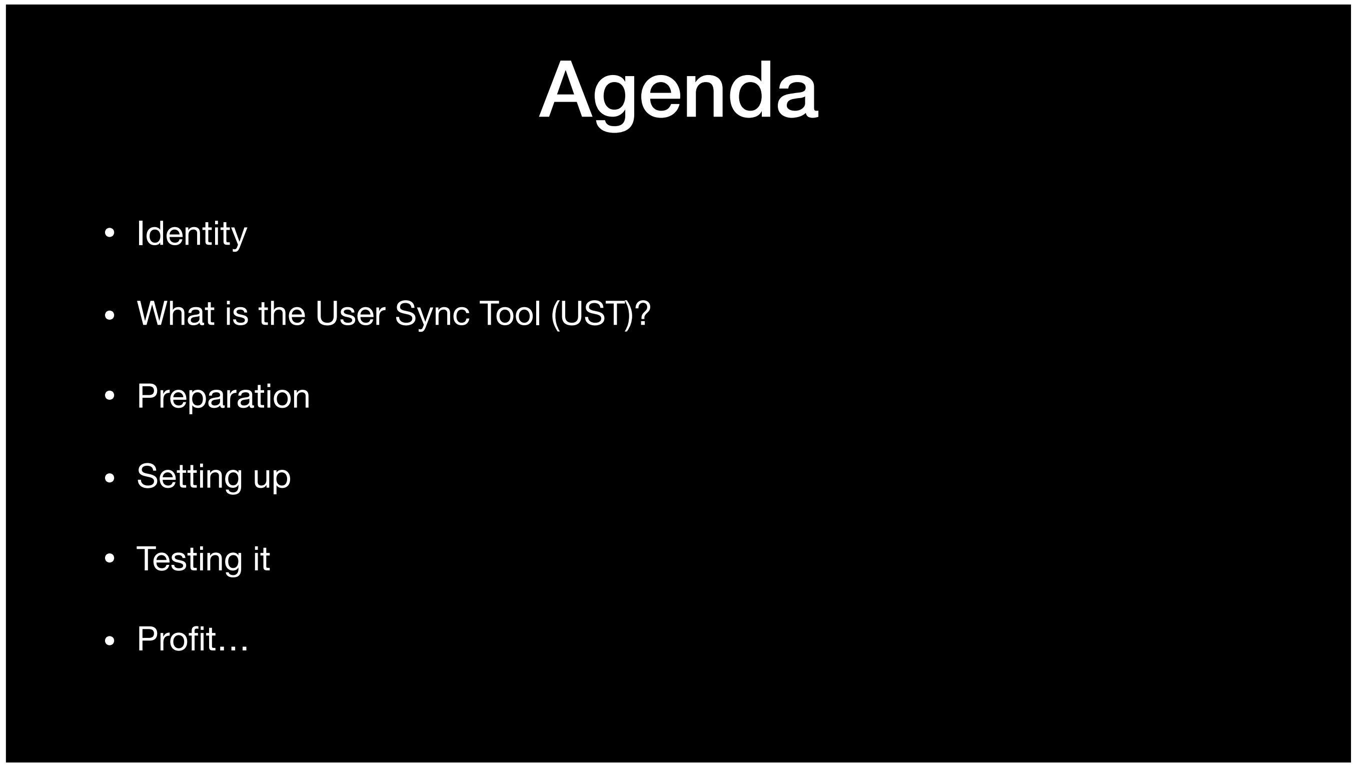
Adobe have been busy - you may have heard about something called Shared Device Licensing. The focus of this talk is more around the User Sync Tool, which may well become something more folks will want to look at because of the introduction of Shared Device Licensing.



SDL Resources

- https://macmule.com/2019/01/30/adobe-shared-device-licensing/
- https://apple.lib.utah.edu/february-2019-macadmins-meeting/
- https://seminars.adobeconnect.com/pmwrvmxilaui/
- https://soundmacguy.wordpress.com/2019/02/01/adobe-shared-device-licensing-answers-to-questions-you-probably-didnt-want-to-ask/

If you want to know more about Shared Device Licensing, check out Ben's blog - he wrote a great summary of it. Darren is also doing a talk for the University of Utah Mac Admins Meeting on the 20th of February. That'll be in the evening. Adobe covered it in a webinar. And I wrote about my experiences when I tried to break it.


Agenda

- Identity
- What is the User Sync Tool (UST)?
- Preparation
- Setting up
- Testing it
- Profit…

So we’re going to go over what identity means in the world of Adobe and the different ways you can deal with it

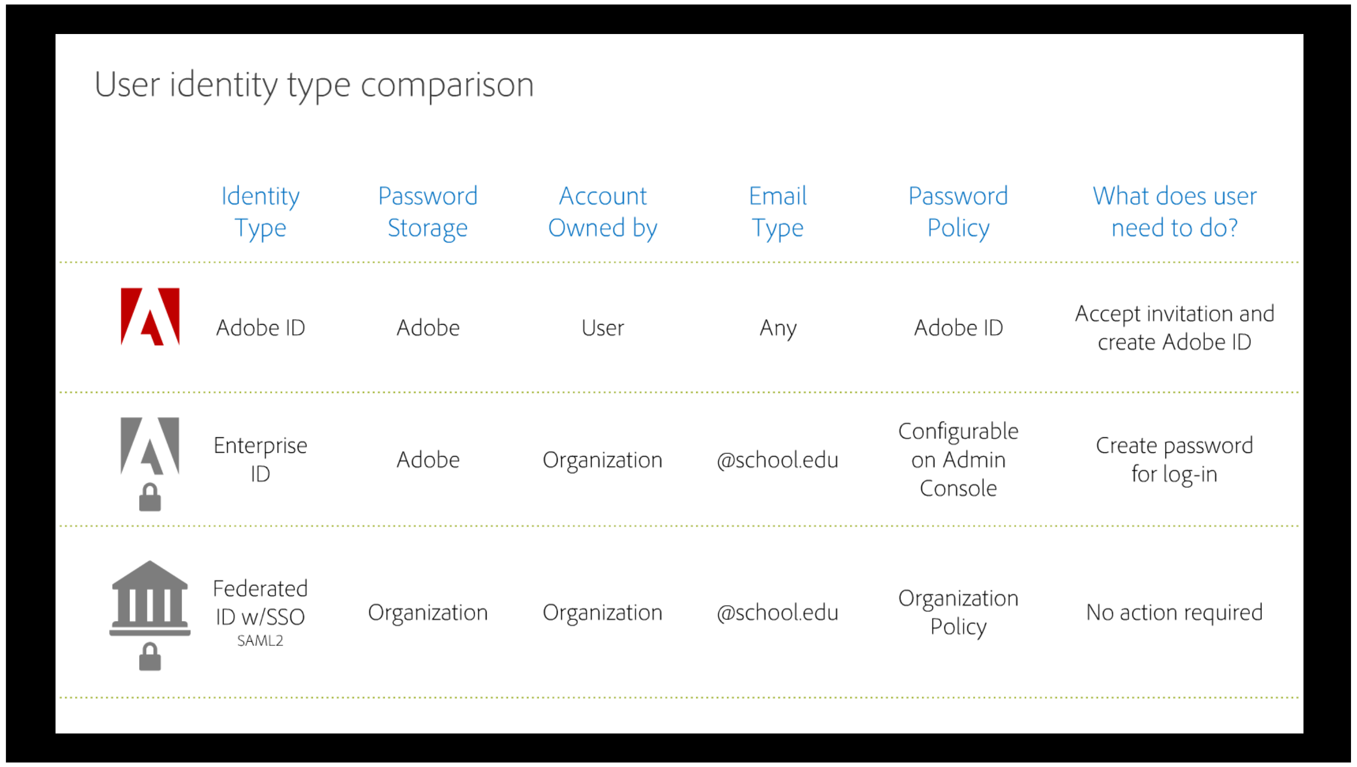
We’ll look at the User Sync Tool, and things to think about when you prepare for it, set it up, and use it, plus some extra considerations and gotchas.

HAWAII
DRIVER LICENSE
01-47-87441
06/03/2008
McLOVIN
MOMONA ST
HONOLULU, HI

HAWAII
DRIVER LICENSE
01-47-87441
06/03/1981
06/03/2008
McLOVIN
892 MOMONA ST
HONOLULU, HI 96820

Identity

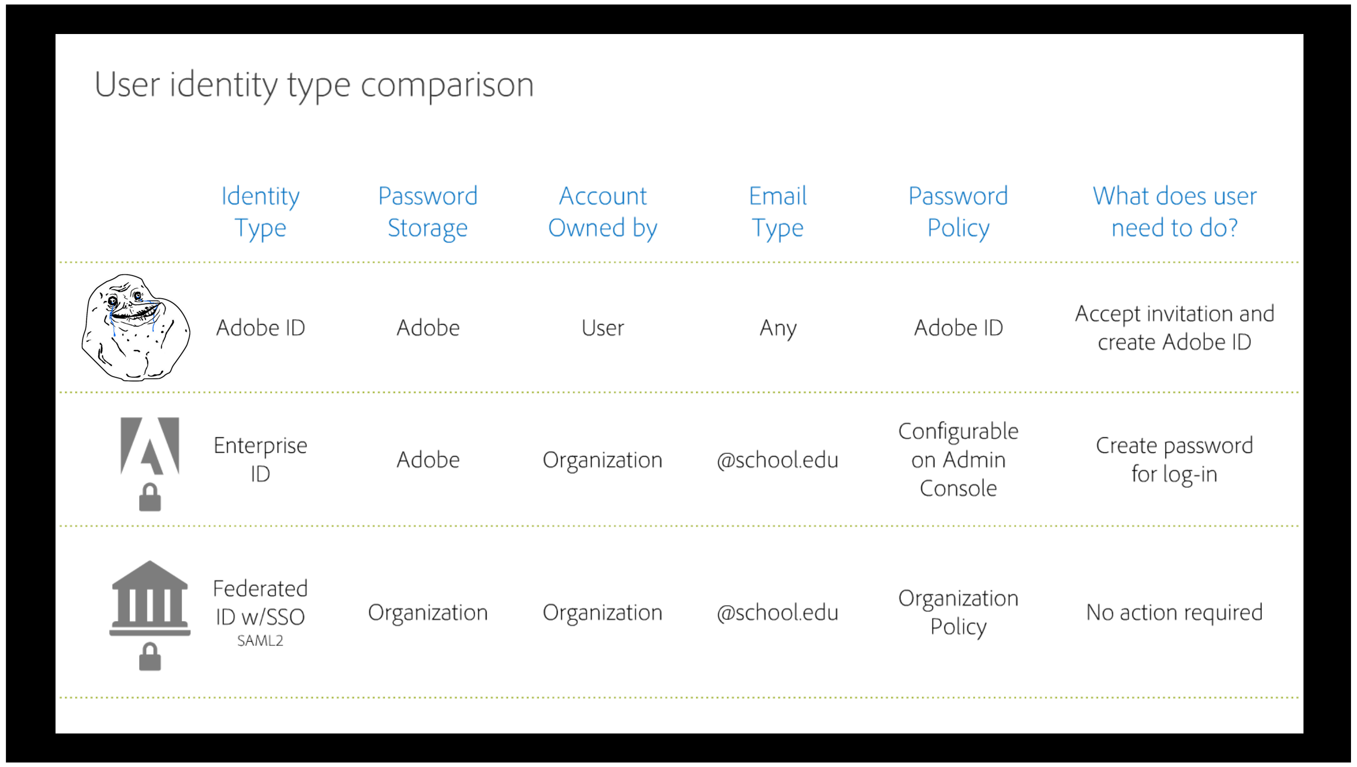
Let’s talk about identity



User identity type comparison

Identity Type	Password Storage	Account Owned by	Email Type	Password Policy	What does user need to do?
Adobe ID	Adobe	User	Any	Adobe ID	Accept invitation and create Adobe ID
Enterprise ID	Adobe	Organization	@school.edu	Configurable on Admin Console	Create password for log-in
Federated ID w/SSO SAML2	Organization	Organization	@school.edu	Organization Policy	No action required

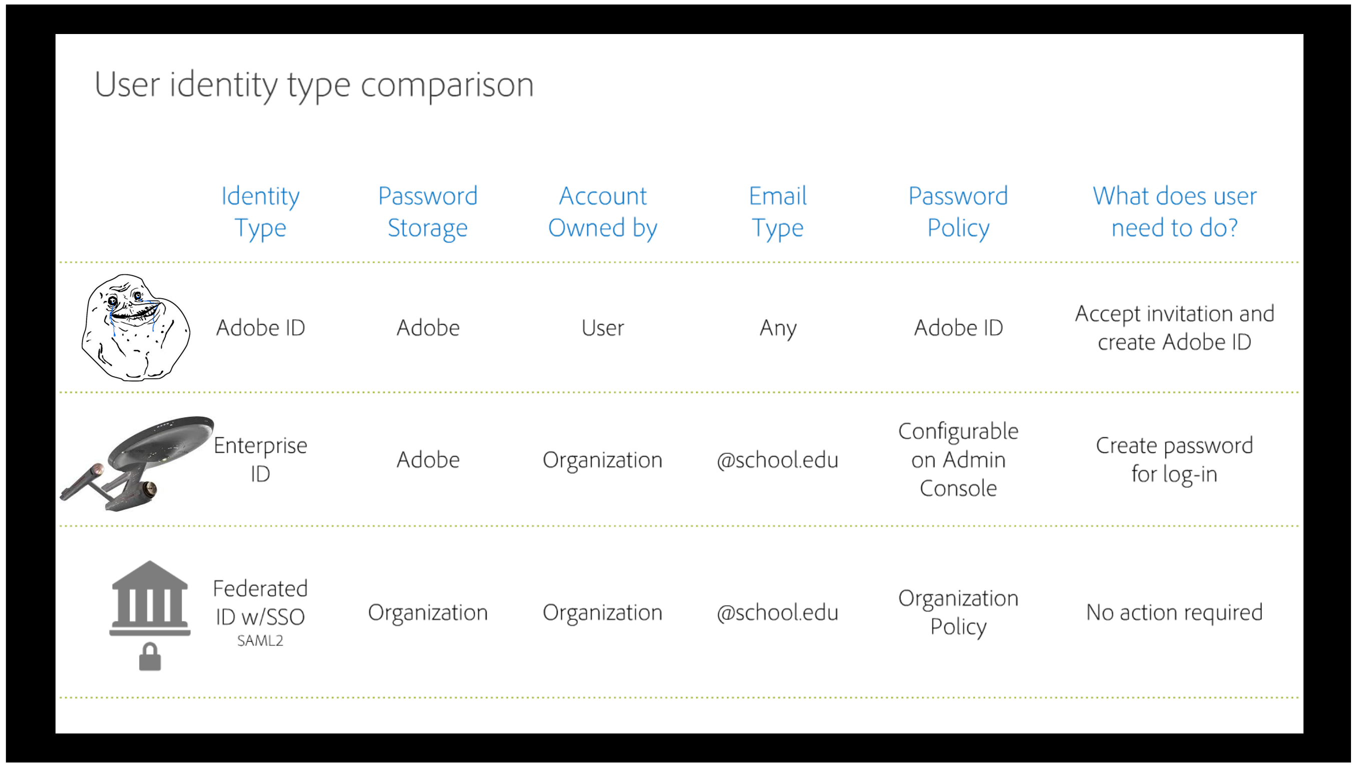
One of the things you'll need to think about now, especially with SDL, is how your users are going to sign into the apps. They need some kind of ID that'll give them access to those apps as well as any named licenses you want to assign to them, which could also include cloud storage. If you were deploying device or serial licensed apps before, chances are your users don't have IDs.



User identity type comparison

	Identity Type	Password Storage	Account Owned by	Email Type	Password Policy	What does user need to do?
	Adobe ID	Adobe	User	Any	Adobe ID	Accept invitation and create Adobe ID
	Enterprise ID	Adobe	Organization	@school.edu	Configurable on Admin Console	Create password for log-in
	Federated ID w/SSO SAML2	Organization	Organization	@school.edu	Organization Policy	No action required

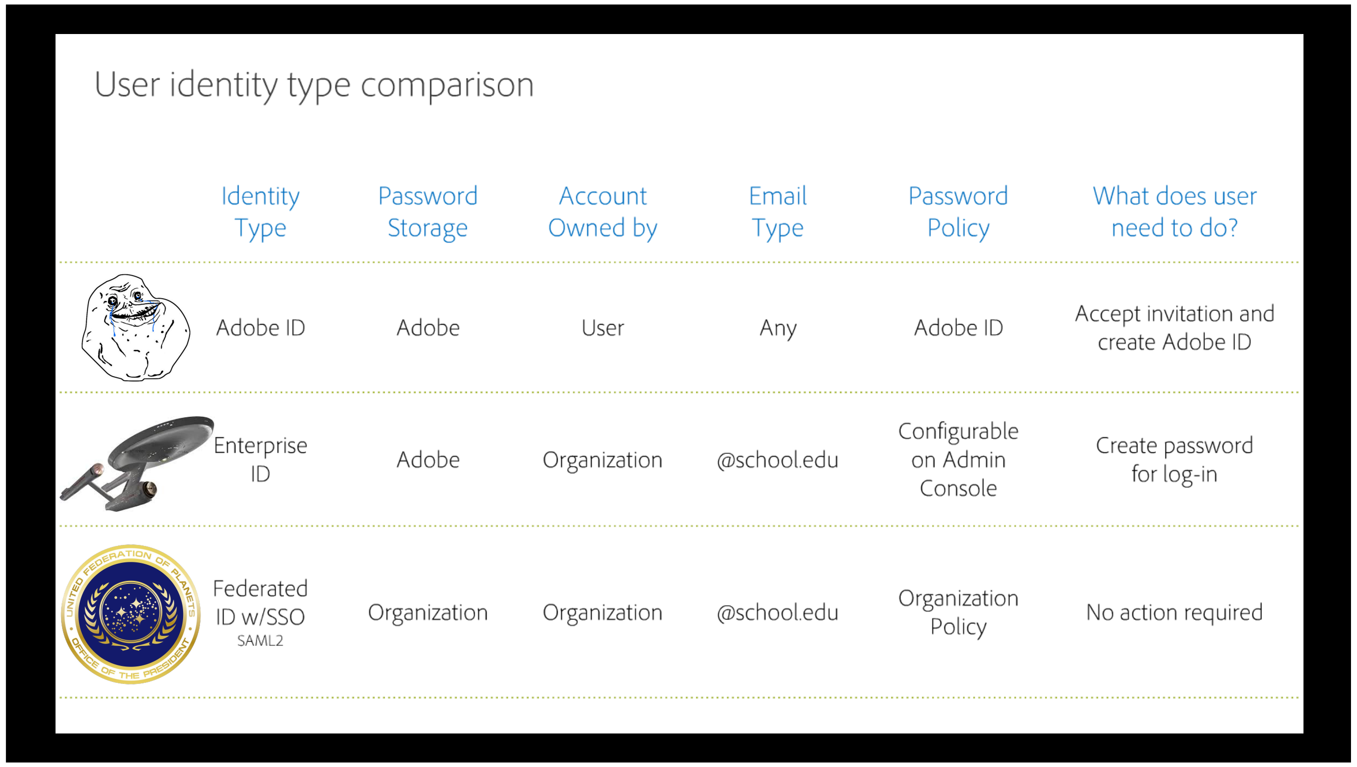
Adobe IDs are basically standalone - they belong to the user and can be tied to any email address. The password and authentication is managed by Adobe. Users can purchase their own product subscriptions and have them tied to their Adobe IDs. You can also grant named licenses to them in your Admin Console.



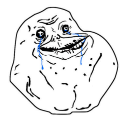
User identity type comparison

	Identity Type	Password Storage	Account Owned by	Email Type	Password Policy	What does user need to do?
	Adobe ID	Adobe	User	Any	Adobe ID	Accept invitation and create Adobe ID
	Enterprise ID	Adobe	Organization	@school.edu	Configurable on Admin Console	Create password for log-in
	Federated ID w/SSO SAML2	Organization	Organization	@school.edu	Organization Policy	No action required

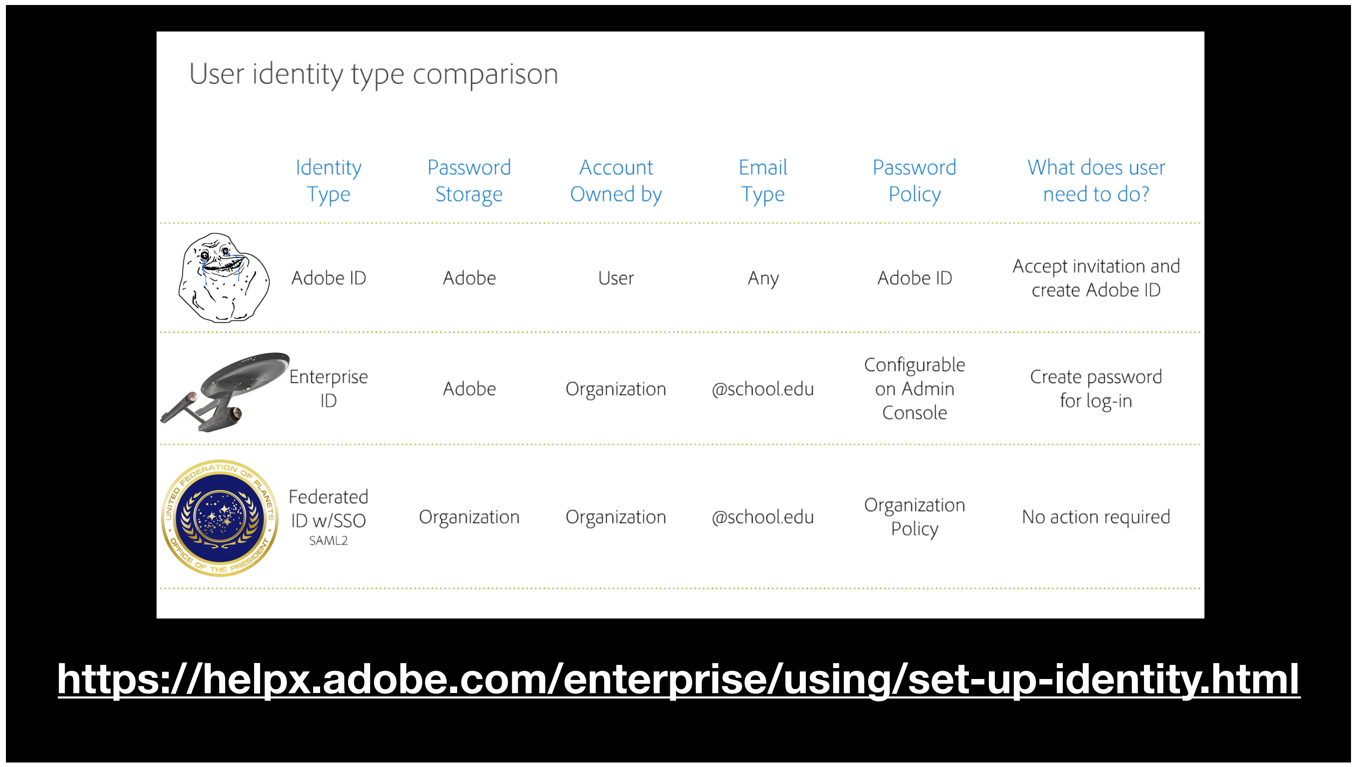
Enterprise IDs - your organisation creates and owns them. They're associated with the user's organisational email address. Passwords and authentication is via Adobe, users need to create and manage separate passwords.



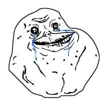
User identity type comparison

	Identity Type	Password Storage	Account Owned by	Email Type	Password Policy	What does user need to do?
	Adobe ID	Adobe	User	Any	Adobe ID	Accept invitation and create Adobe ID
	Enterprise ID	Adobe	Organization	@school.edu	Configurable on Admin Console	Create password for log-in
	Federated ID w/SSO SAML2	Organization	Organization	@school.edu	Organization Policy	No action required

Federated IDs - they're the same as enterprise IDs but authentication is done through your IdP - so it's SSO over SAML. These take some more setting up at your end but if you're in a place where your users have directory accounts, they'll really get the best experience.



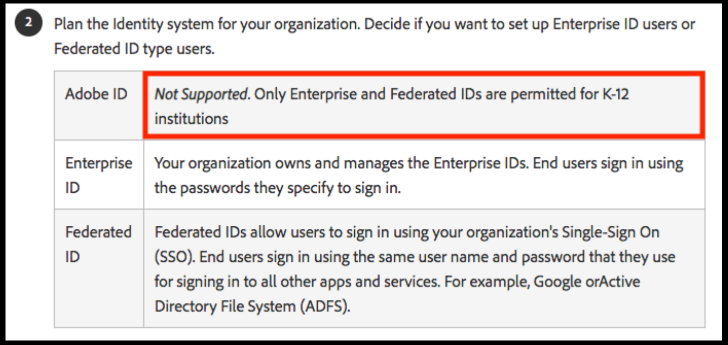
User identity type comparison

	Identity Type	Password Storage	Account Owned by	Email Type	Password Policy	What does user need to do?
	Adobe ID	Adobe	User	Any	Adobe ID	Accept invitation and create Adobe ID
	Enterprise ID	Adobe	Organization	@school.edu	Configurable on Admin Console	Create password for log-in
	Federated ID w/SSO SAML2	Organization	Organization	@school.edu	Organization Policy	No action required

https://helpx.adobe.com/enterprise/using/set-up-identity.html

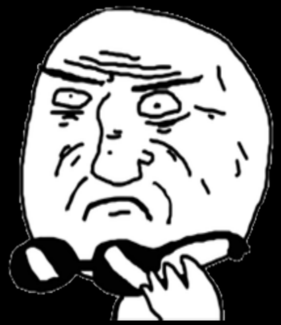
If you’re not using Enterprise or Federated IDs and you want to, you need to set up your domain and directory in your Admin Console - follow the instructions there.

Oh yeah…



2 Plan the Identity system for your organization. Decide if you want to set up Enterprise ID users or Federated ID type users.

Adobe ID	*Not Supported.* Only Enterprise and Federated IDs are permitted for K-12 institutions
Enterprise ID	Your organization owns and manages the Enterprise IDs. End users sign in using the passwords they specify to sign in.
Federated ID	Federated IDs allow users to sign in using your organization's Single-Sign On (SSO). End users sign in using the same user name and password that they use for signing in to all other apps and services. For example, Google orActive Directory File System (ADFS).

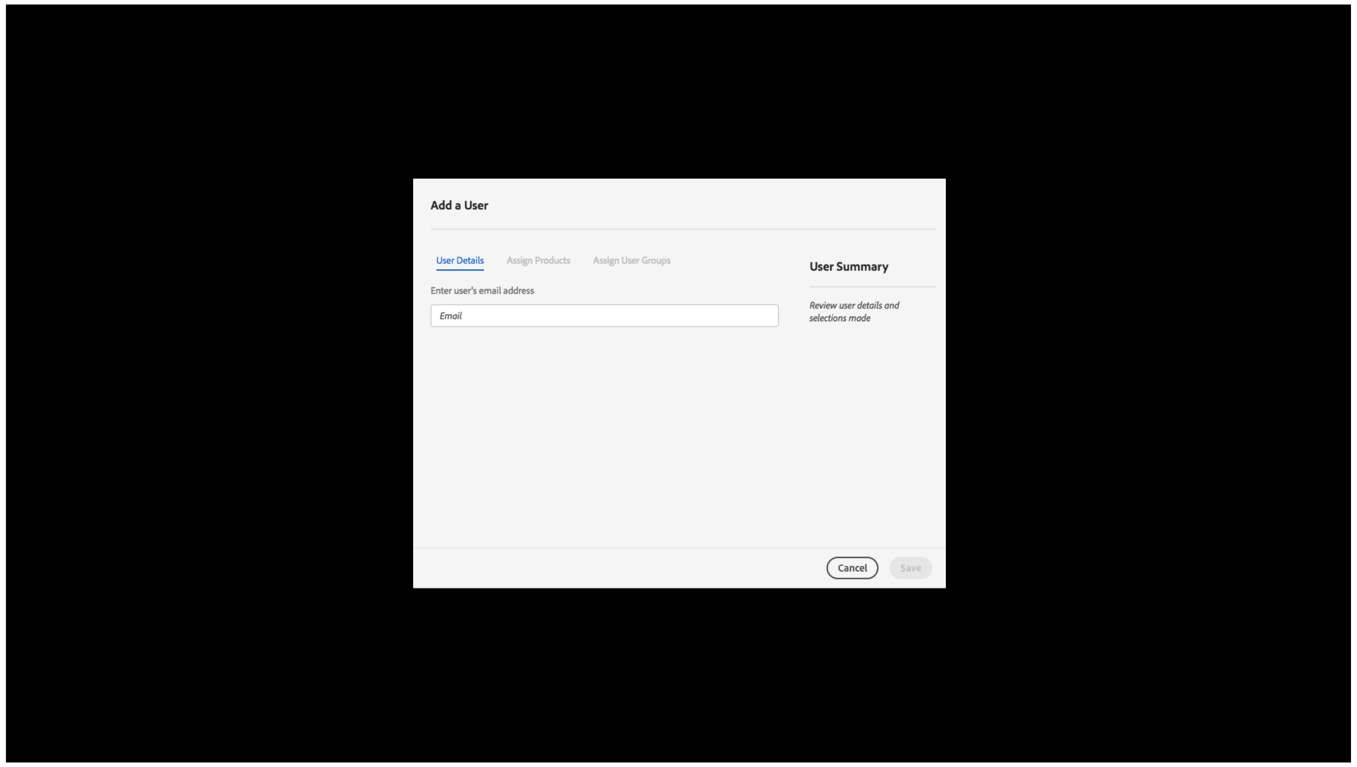


https://helpx.adobe.com/enterprise/using/sdl-deployment-guide.html

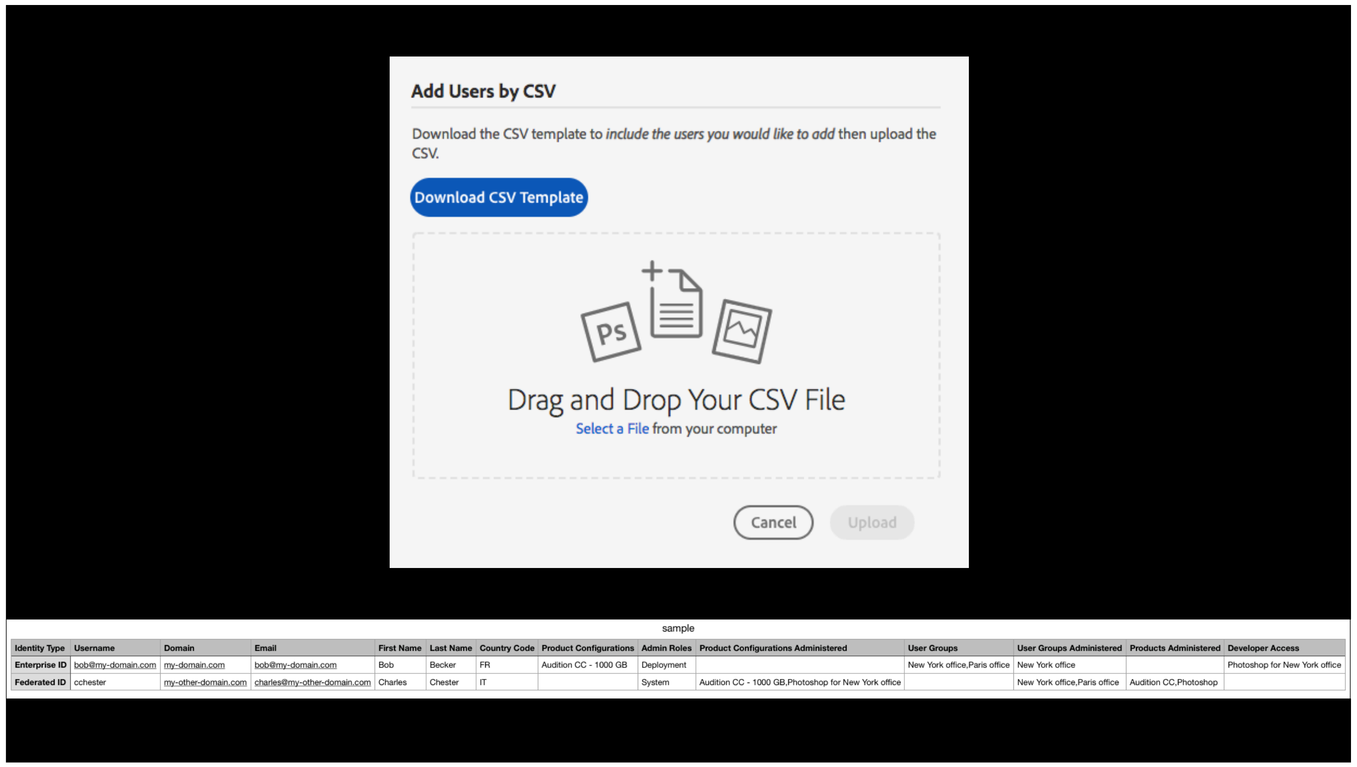
I saw this in the Shared Device License deployment guide… If you're in a school, you might _have_ to use Enterprise or Federated IDs… I'm not 100% sure if this applies outside the US and I don't work in a school so can't really say much else - but I'd say get in touch with your reseller or Adobe rep - I've found the support area of the Admin Console really helpful.

Go forth and create…

So now you’re going to create some IDs in your admin console. You’ve got a few ways…



You can artisinally create each user, one by one, then assign their groups and product profiles (named licenses) by hand.



Add Users by CSV

Download the CSV template to *include the users you would like to add* then upload the CSV.

sample

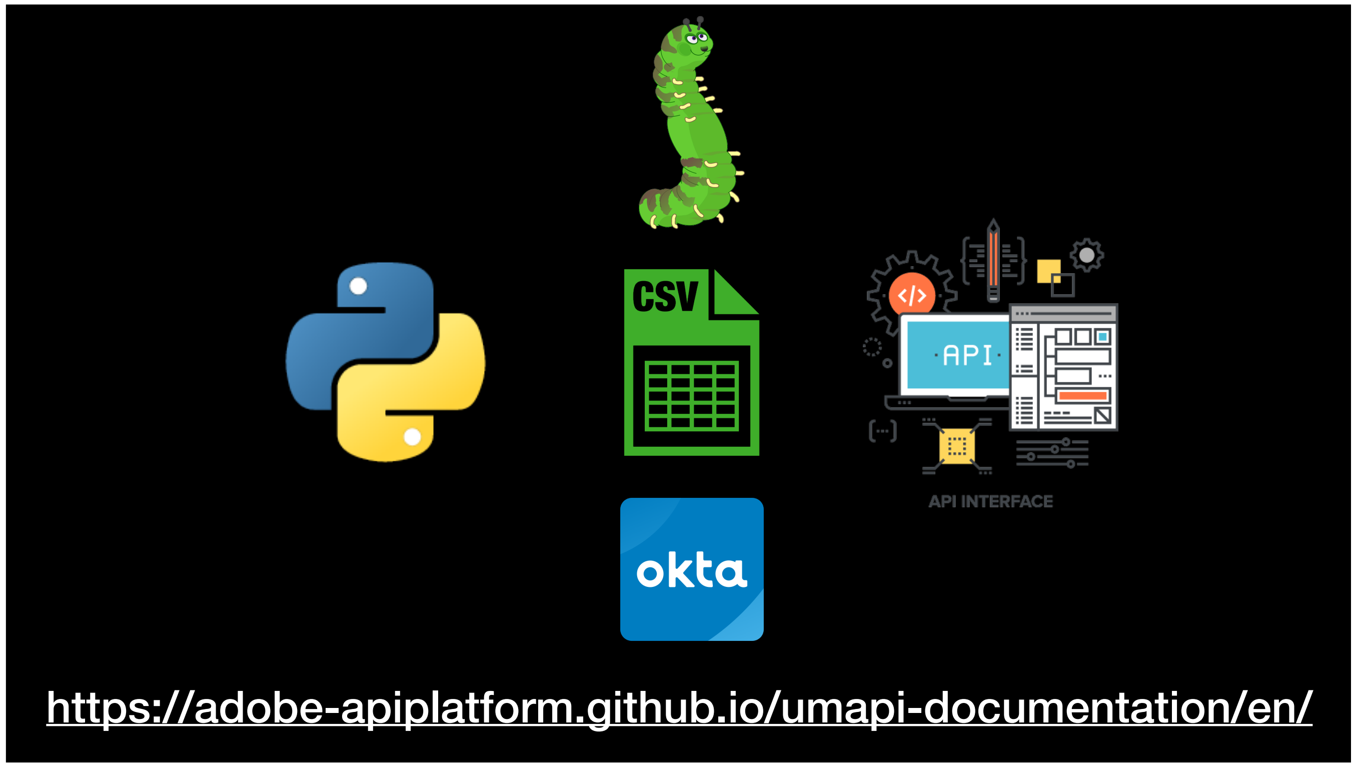
Identity Type	Username	Domain	Email	First Name	Last Name	Country Code	Product Configurations	Admin Roles	Product Configurations Administered	User Groups	User Groups Administered	Products Administered	Developer Access
Enterprise ID	bob@my-domain.com	my-domain.com	bob@my-domain.com	Bob	Becker	FR	Audition CC - 1000 GB	Deployment		New York office,Paris office	New York office		Photoshop for New York office
Federated ID	cchester	my-other-domain.com	charles@my-other-domain.com	Charles	Chester	IT		System	Audition CC - 1000 GB,Photoshop for New York office		New York office,Paris office	Audition CC,Photoshop	

Import a CSV - still a bit manual - but at least you can assign groups and licenses in one fell swoop.

But then you'll need to repeat the process regularly to deal with new people and those that have left.

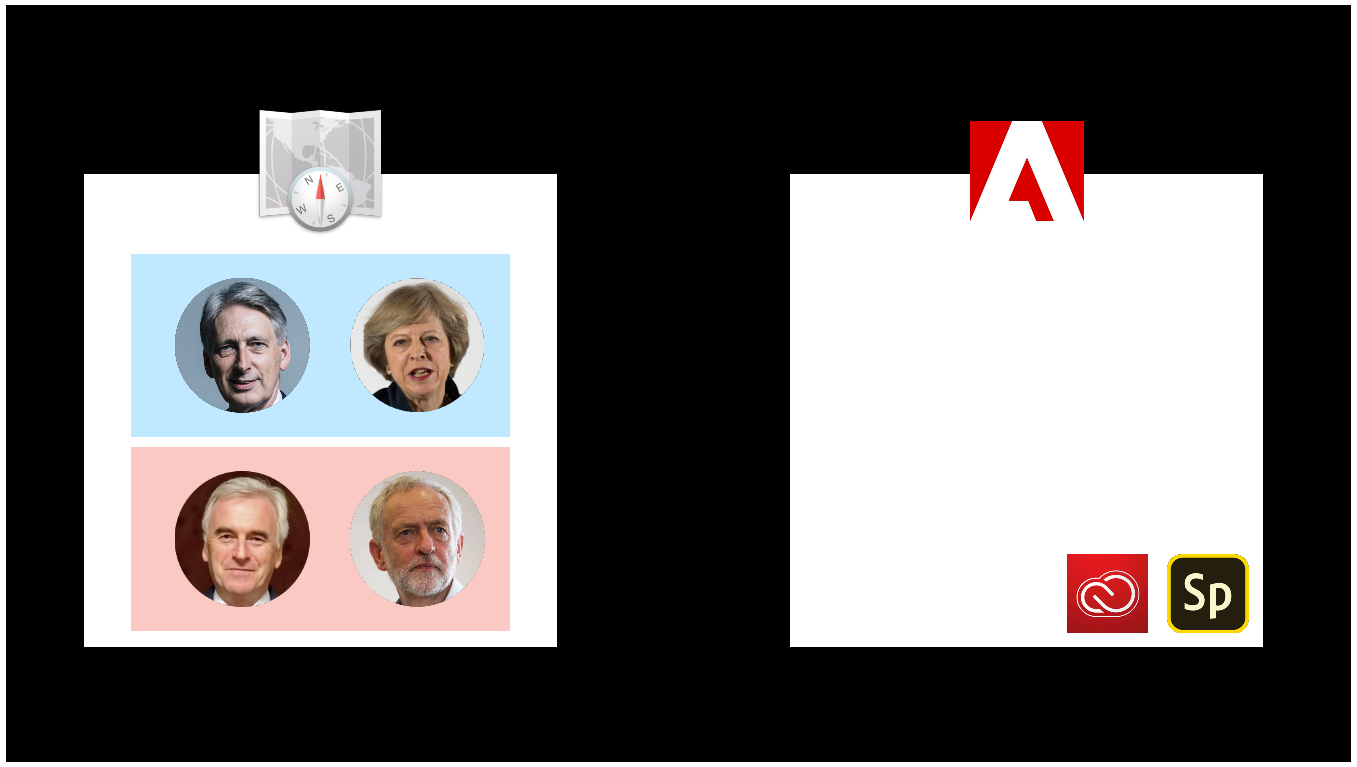
The User Sync Tool (UST)

So that brings us to the User Sync Tool…

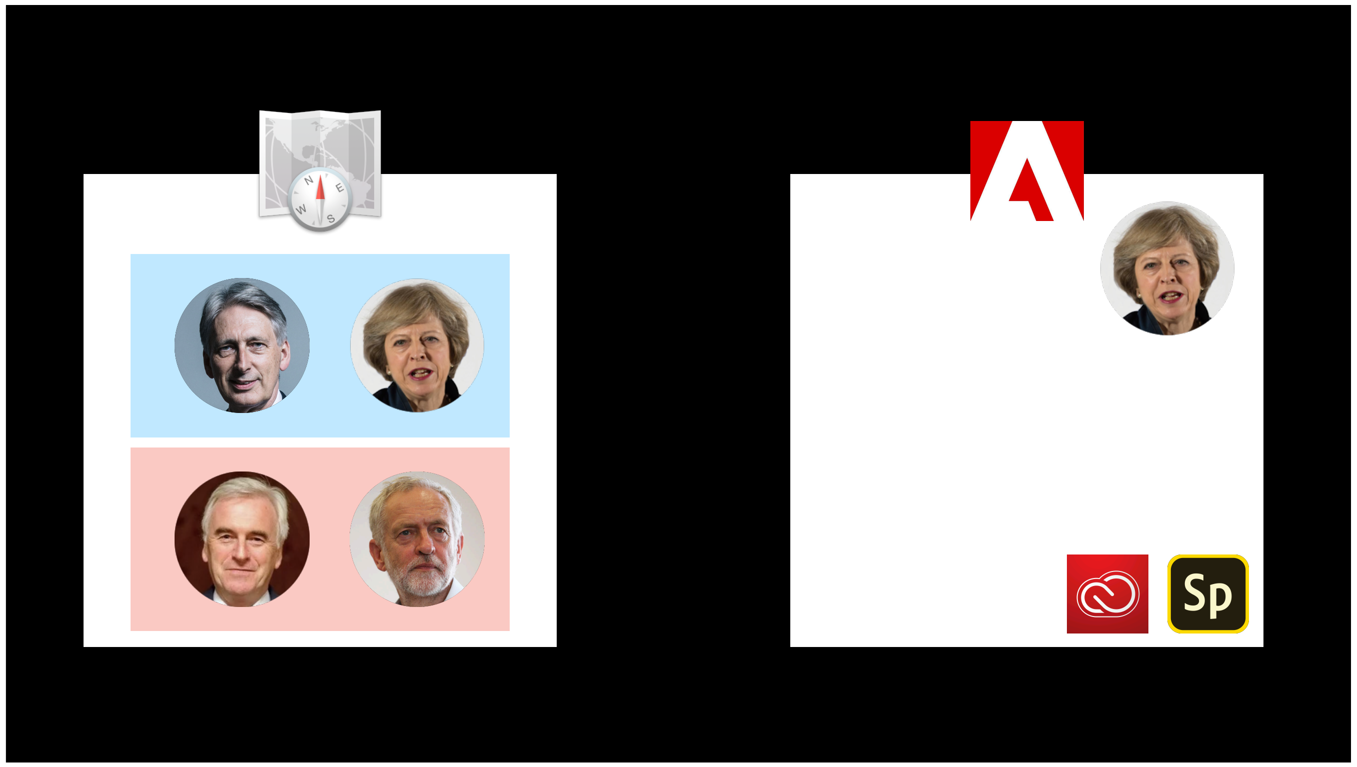


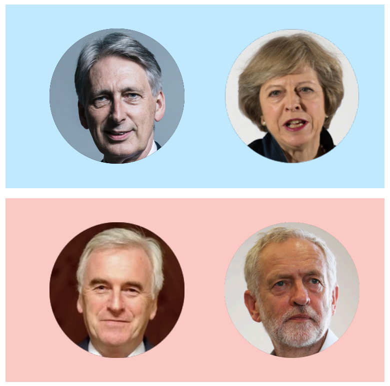
https://adobe-apiplatform.github.io/umapi-documentation/en/

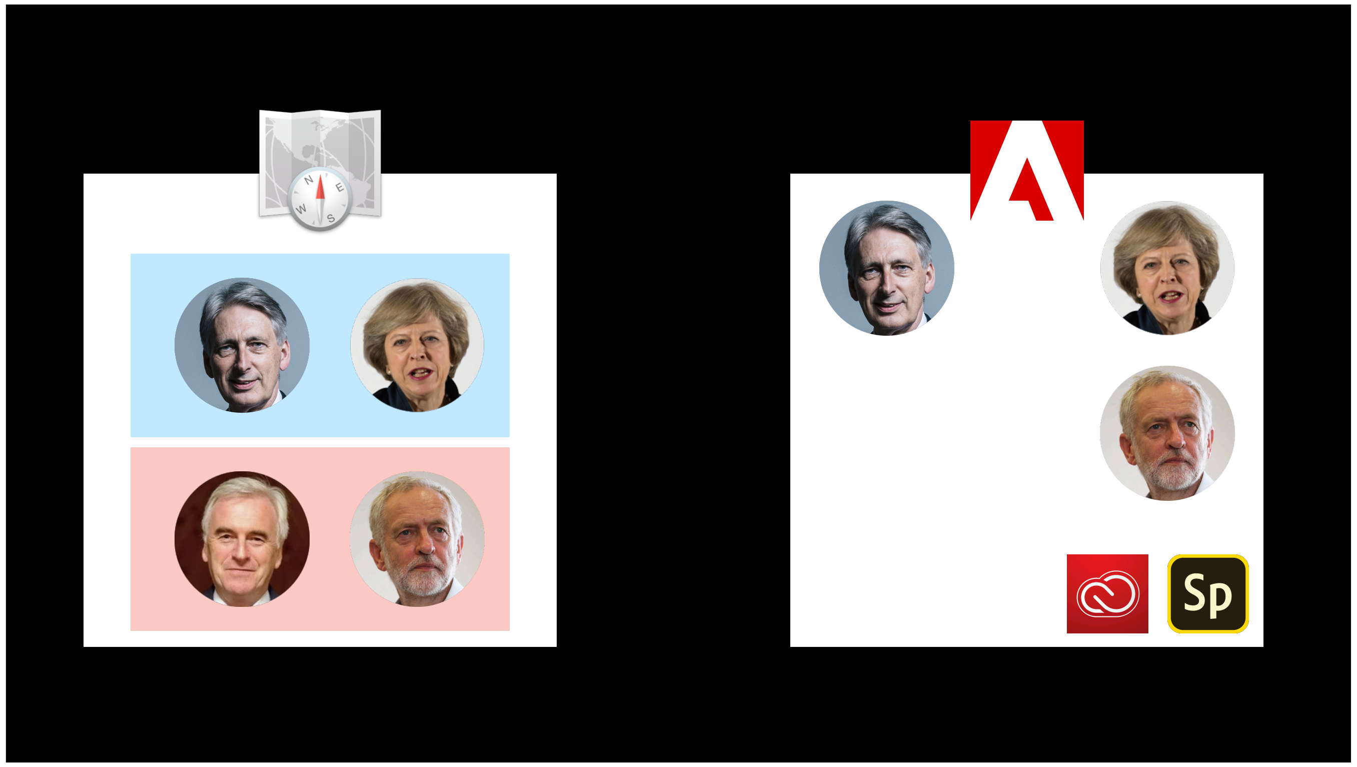
It’s a Python script you run on one of your computers (probably a server). It reads users from a source such as an LDAP directory service, CSV file or OKTA. Then it does things to user IDs in your Admin Console using Adobe’s User Management API, which you can read all about here…

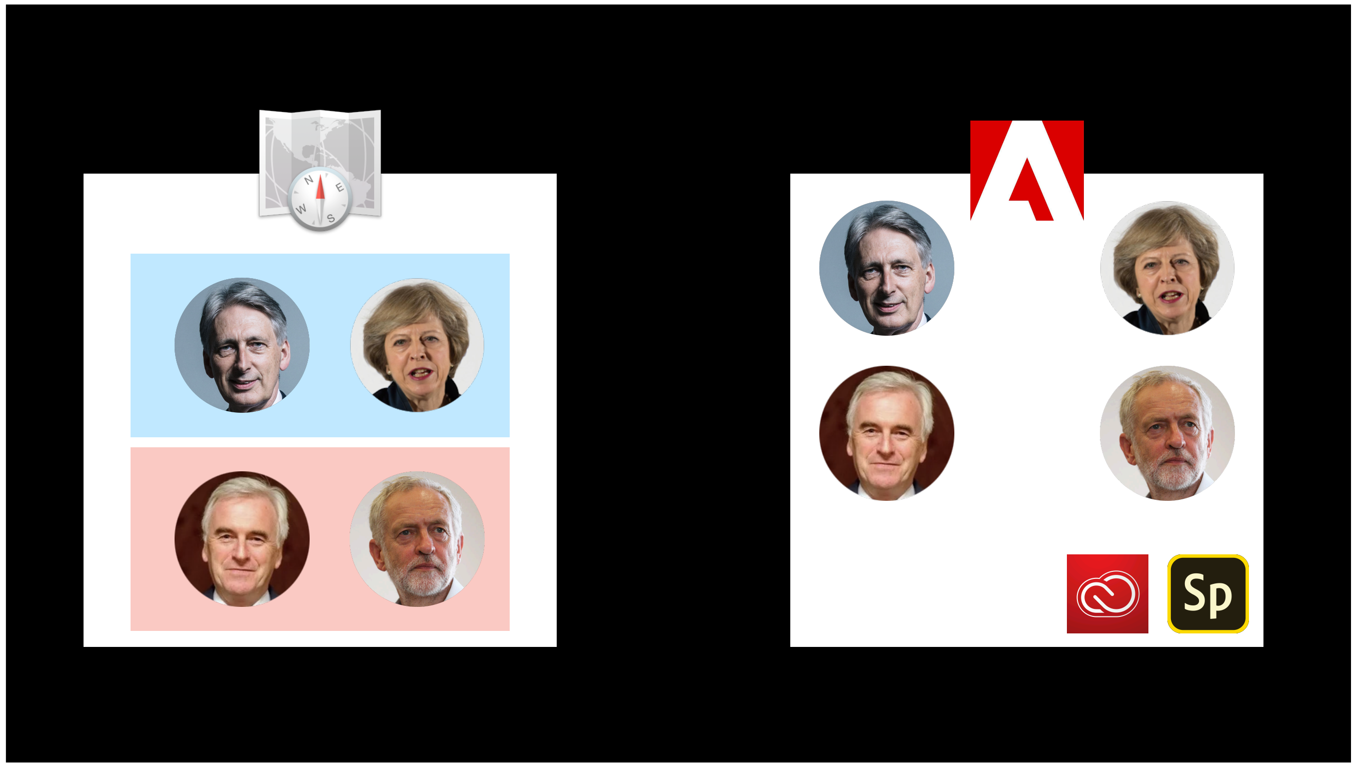


It can create identities for your users in your Admin Console…

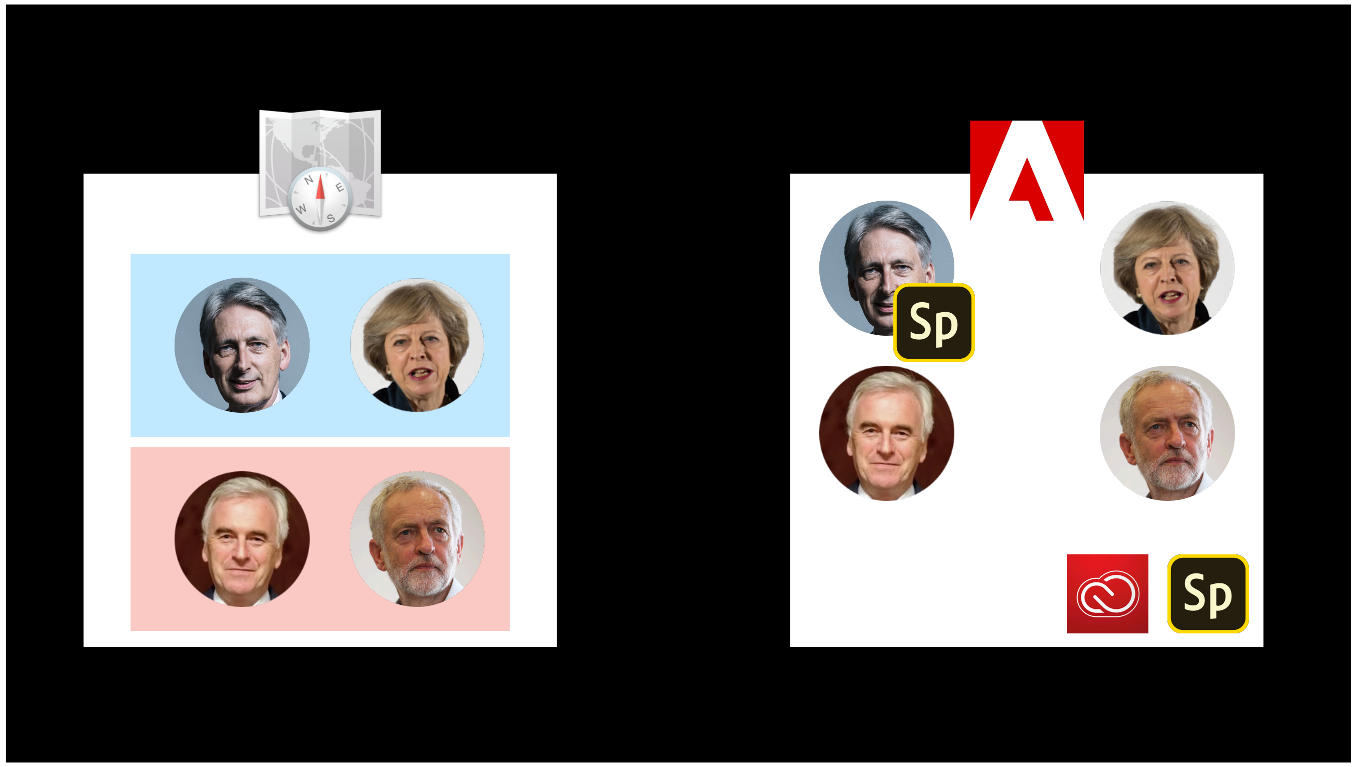


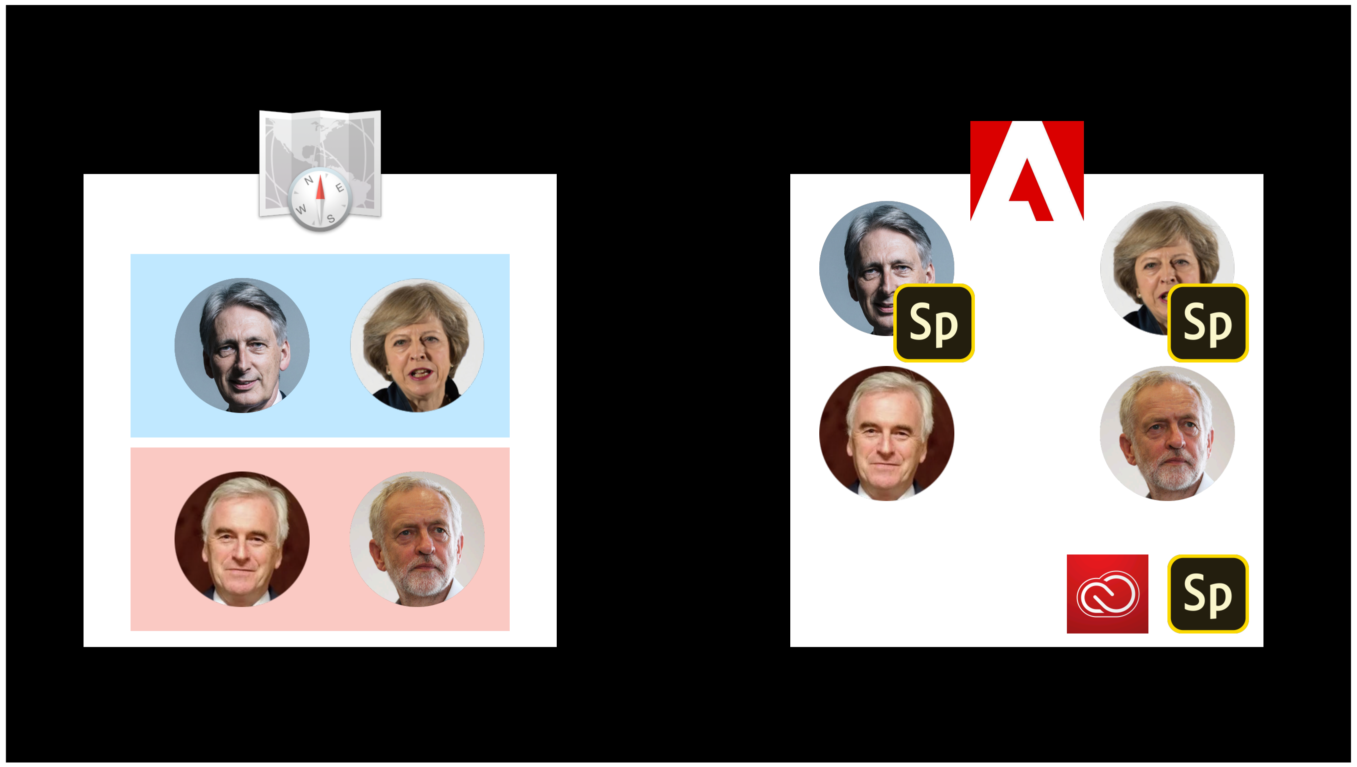


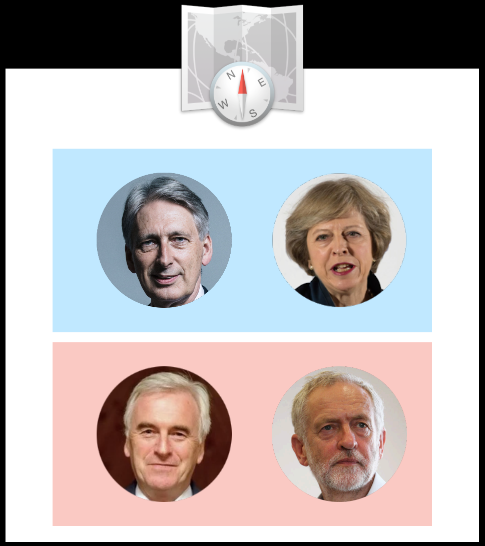


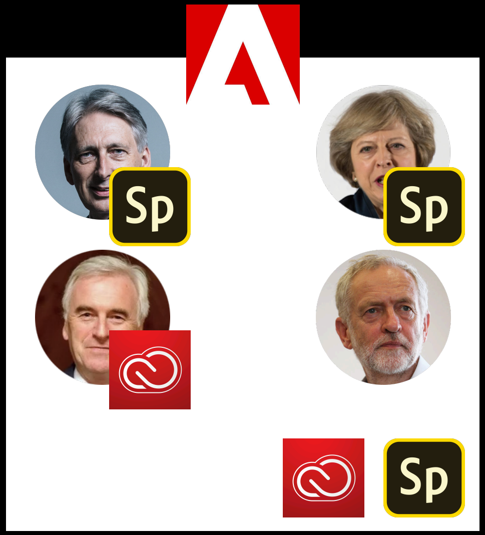


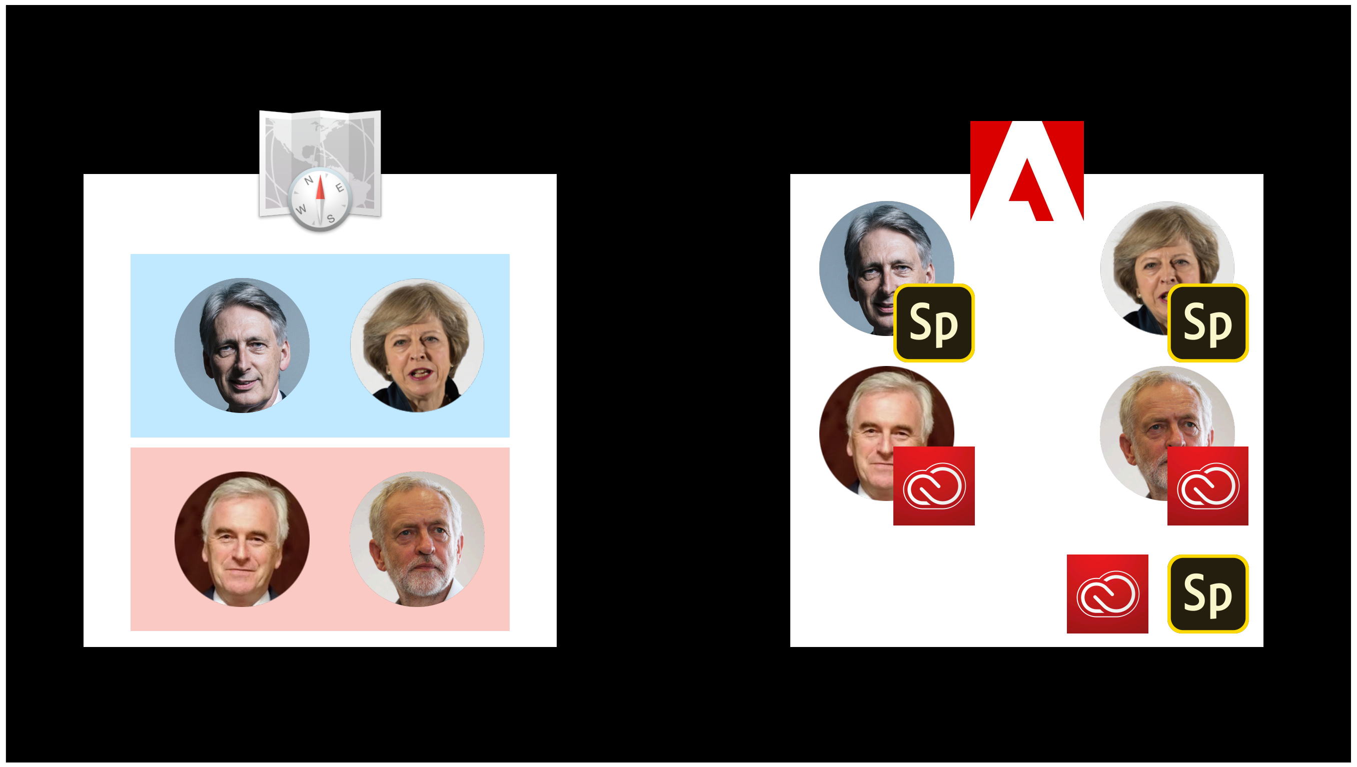
And assign them product profiles, or licenses.



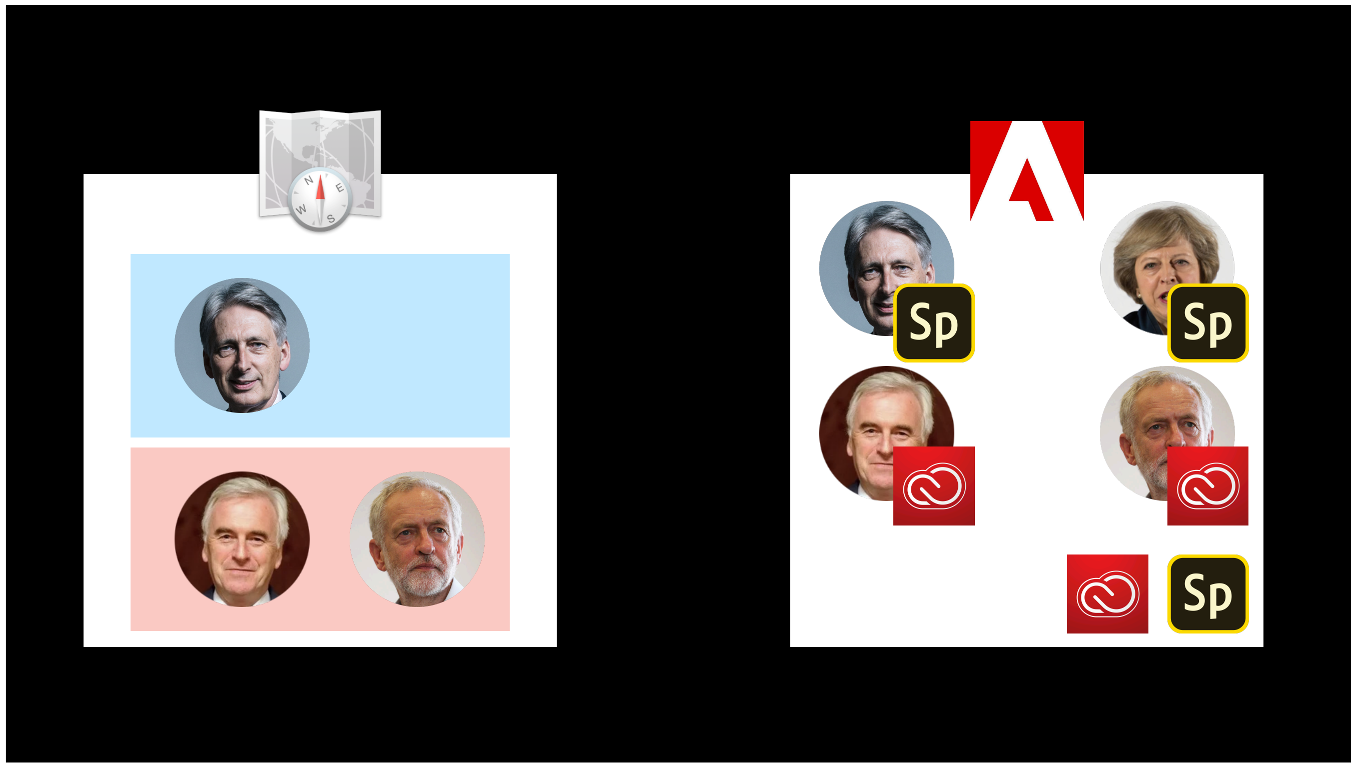




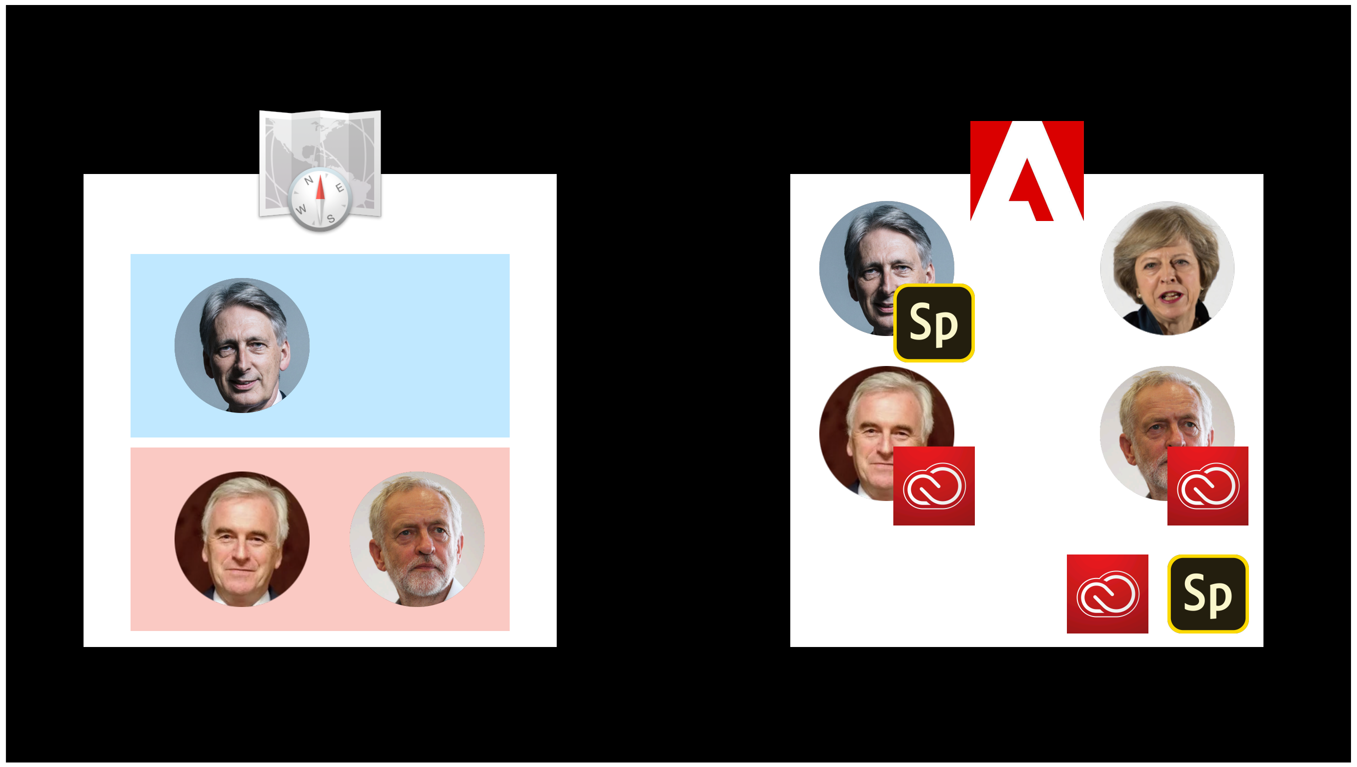

Sp
Sp
Sp



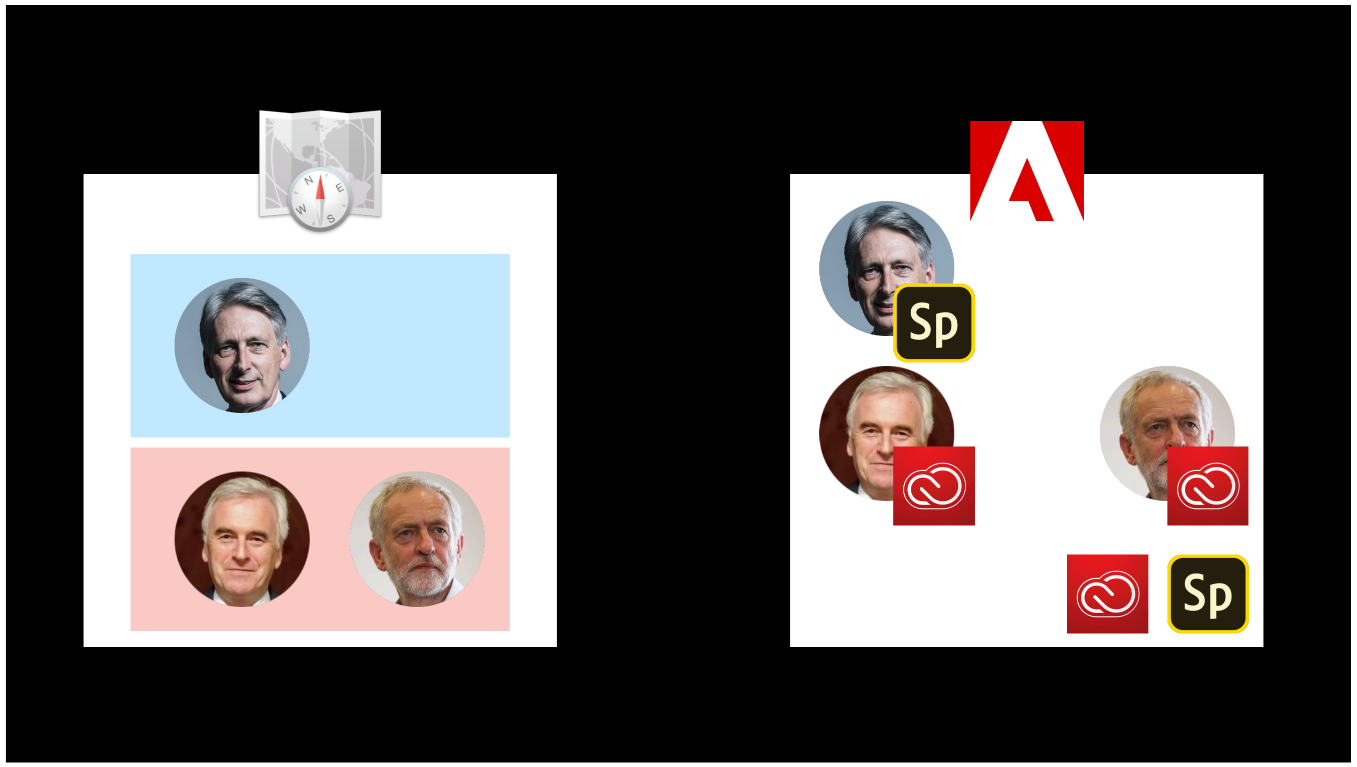
If a user leaves your organisation…



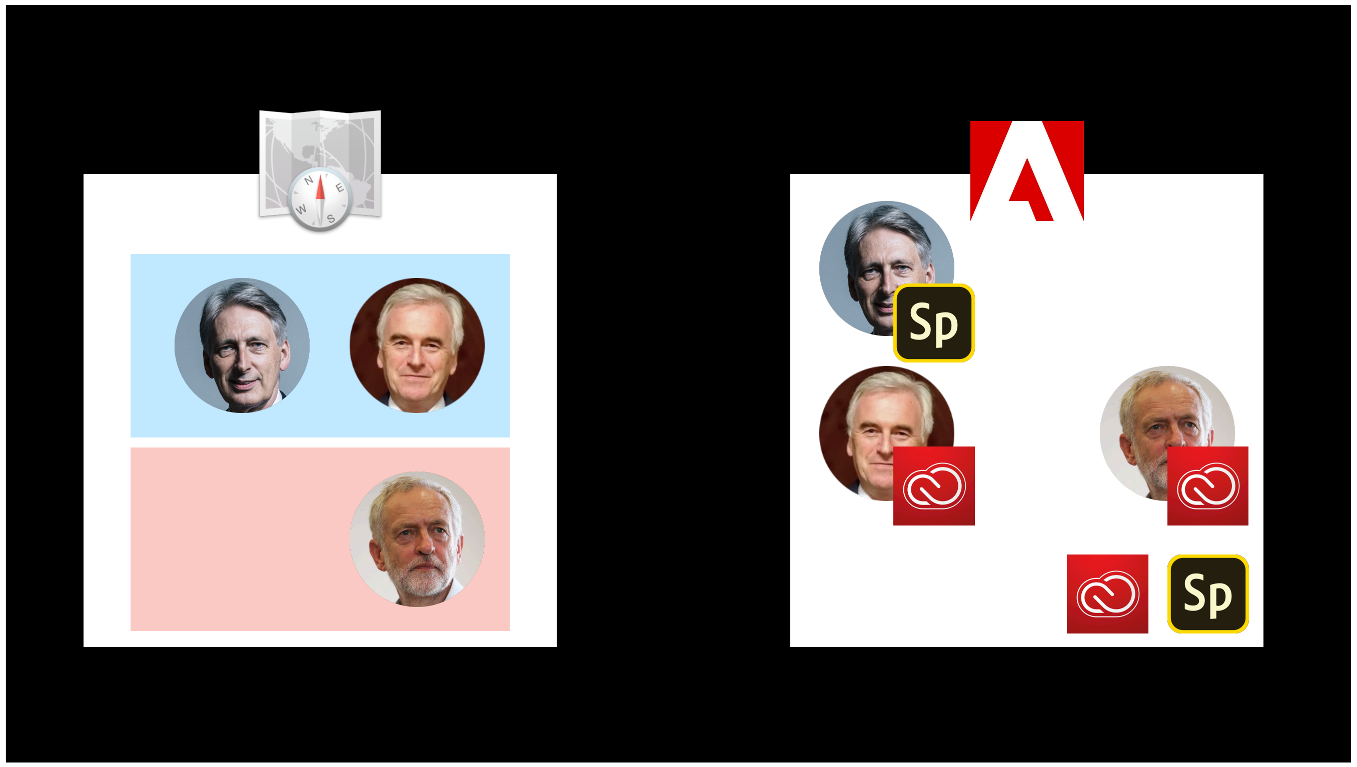
If a user leaves your organisation…



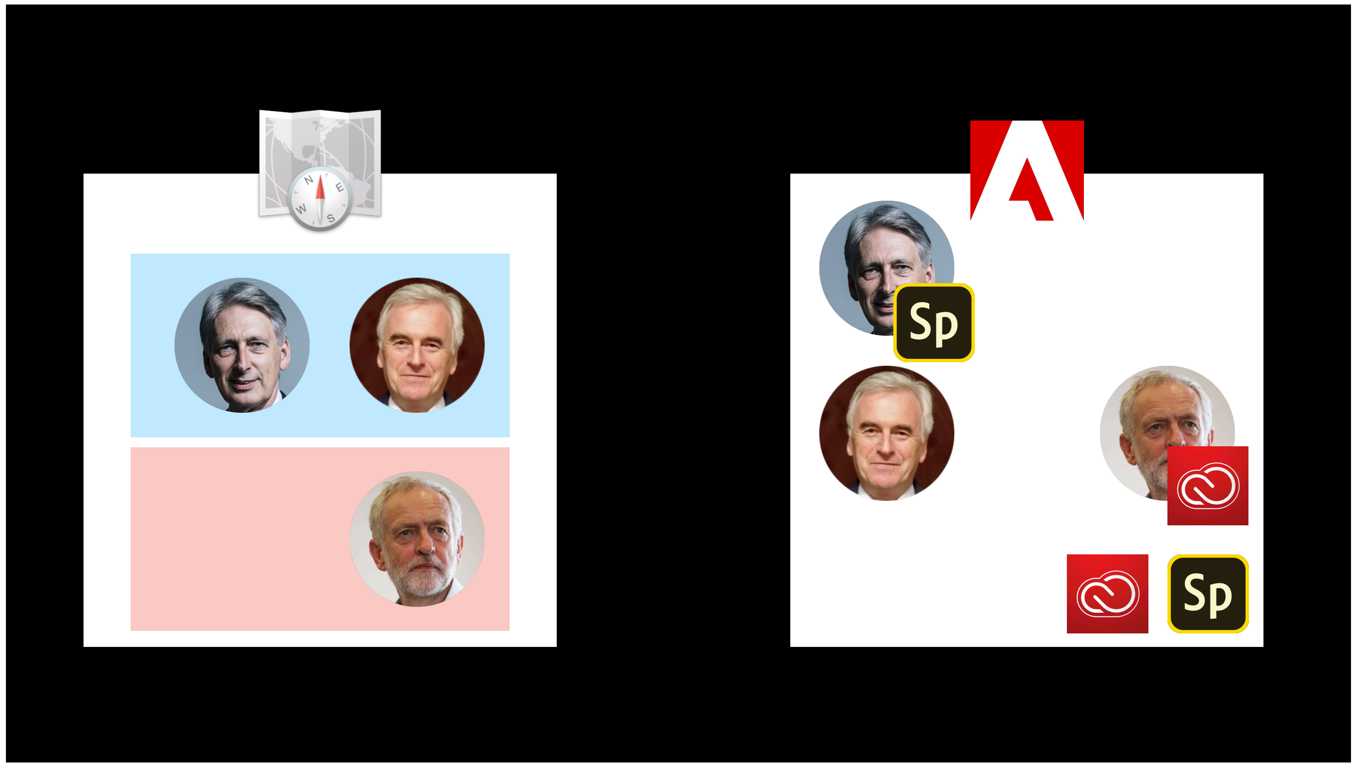
The UST can, if you like, remove their product profiles - no more licenses for you, Prime Minister…



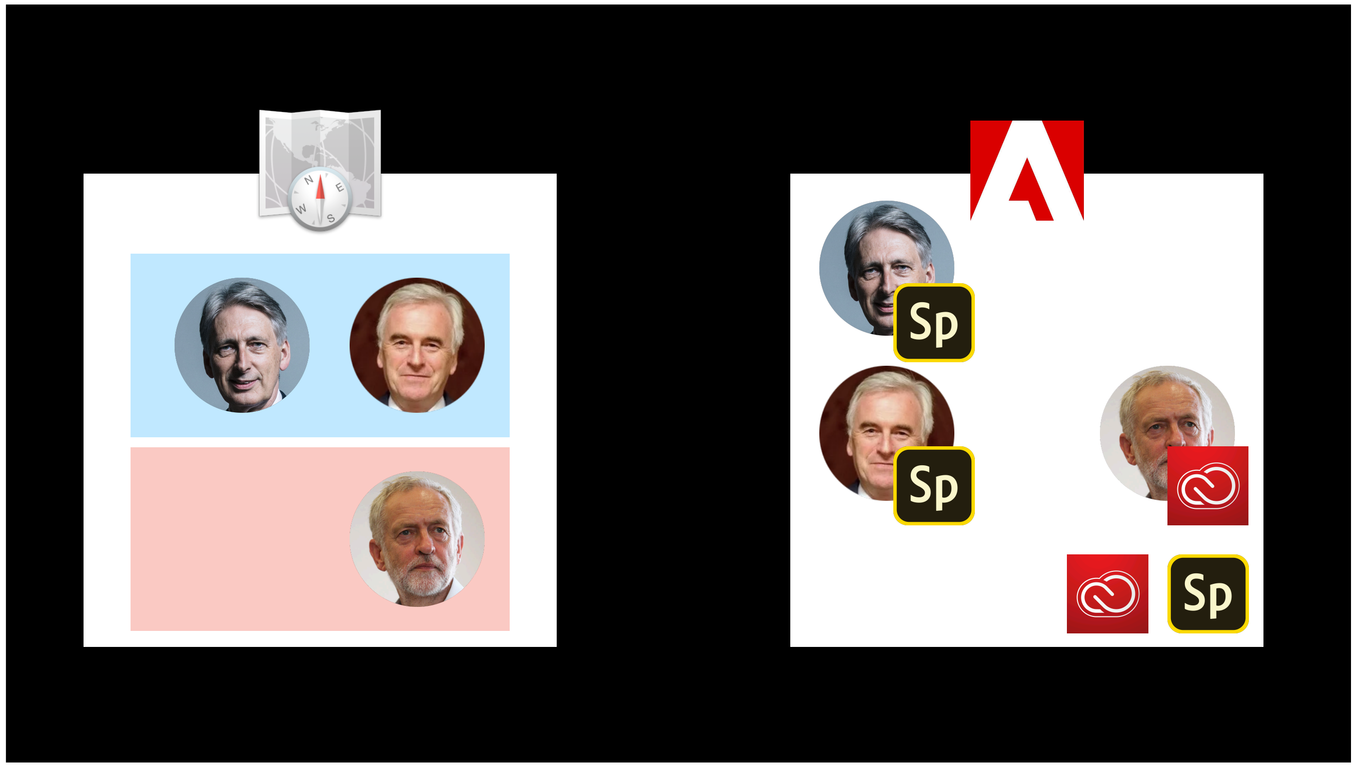
And, if you like, the user account in the Admin Console too…

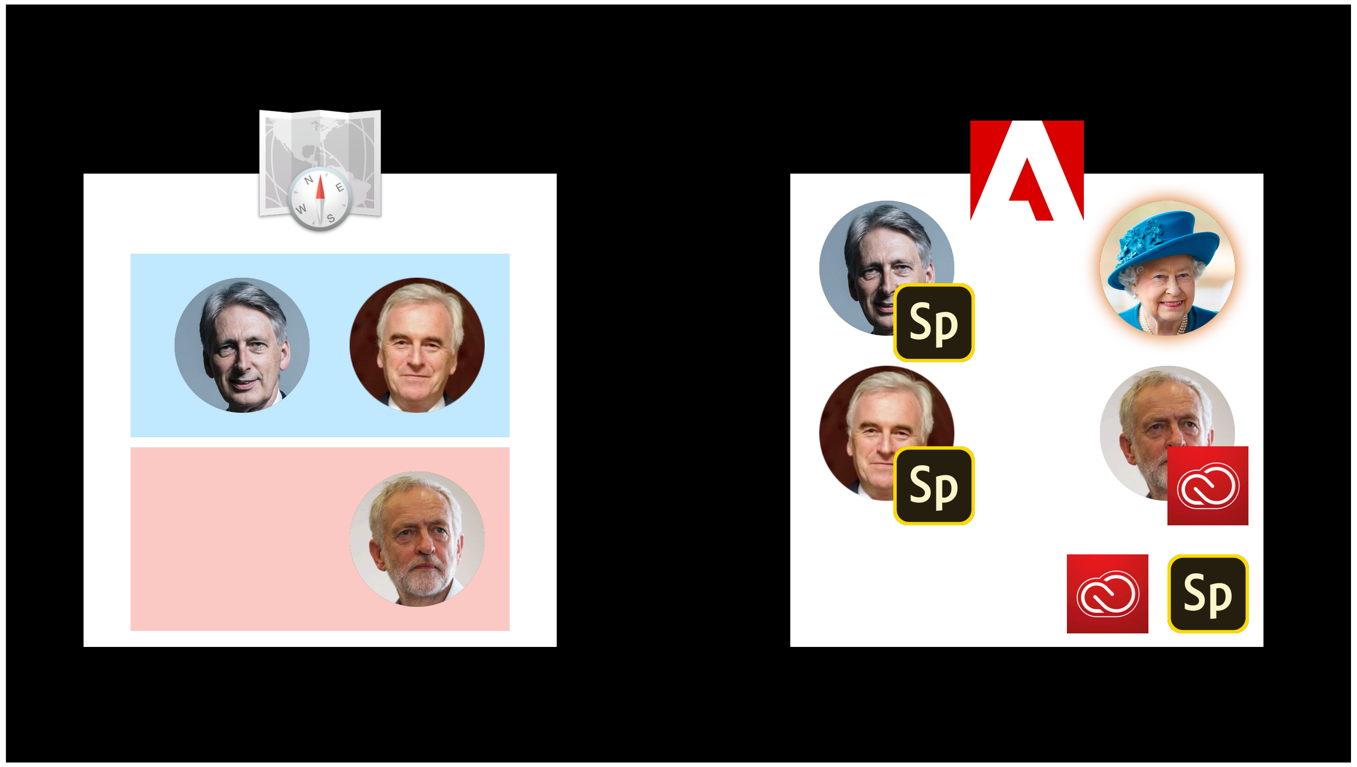


If the user’s group membership changes…



The UST updates their product assignments…


Sp
Sp
Sp



And it can also exclude or ignore IDs in your Admin Console by identity type (i.e. Adobe ID, Enterprise ID, Federated ID) group assignment in your Admin Console, or by regular expression matching the username/email address. This is useful if you have, say, an Adobe ID set up as a system administrator in the Admin Console, which is a good idea if you're using Federated IDs and something goes wrong with the SSO integration.

AllFunnyPictures.com

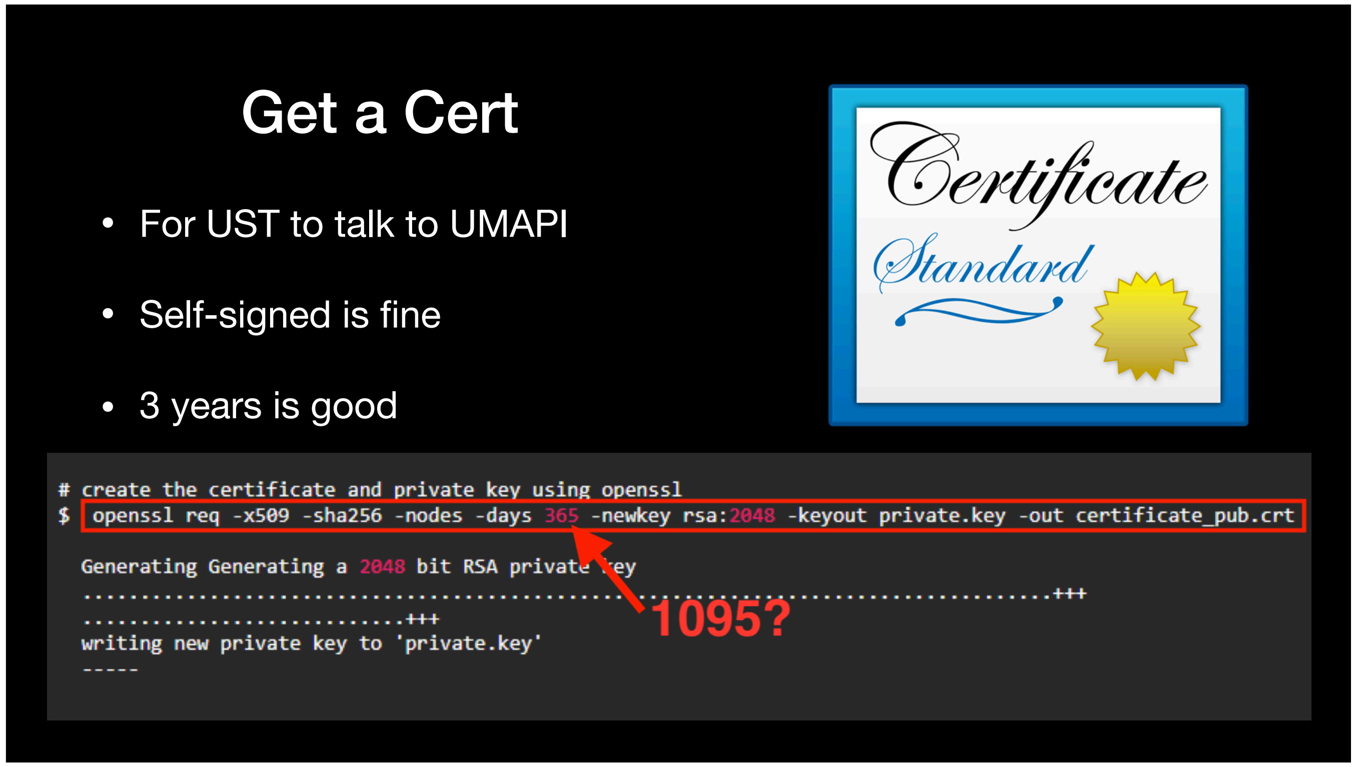
Preparation

Read up!

- Set up the User Sync Tool:
  - https://helpx.adobe.com/uk/enterprise/using/user-sync.html
- Setup and Success Guide:
  - https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/

Have a look through the documentation - the first link covers the basics of what you need to do, but the Setup and Success guide is a lot more in-depth - it covers more scenarios and has helpful hints around a lot of the best practices you should be thinking about when you set this thing up.

Everything I'm going to talk about from now on is covered in these documents.



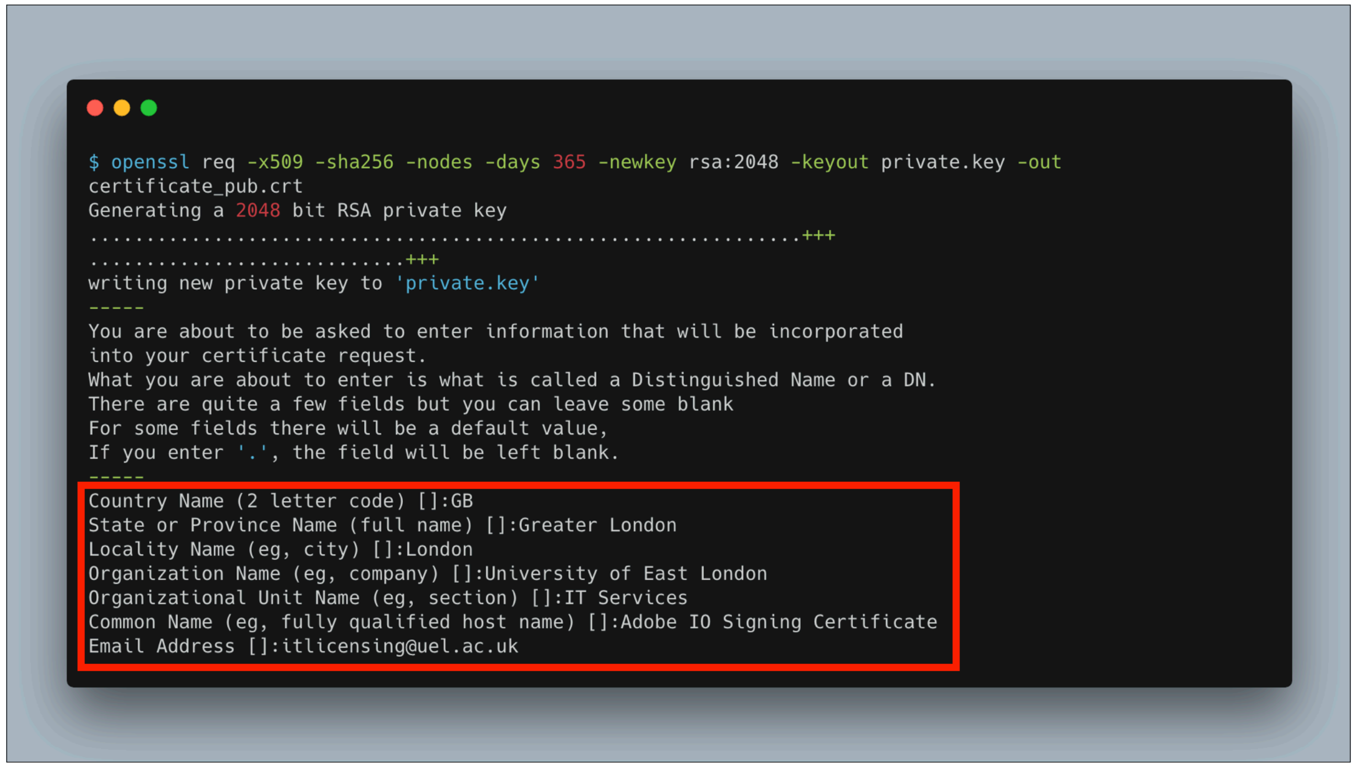
Get a Cert

- For UST to talk to UMAPI
- Self-signed is fine
- 3 years is good

```
# create the certificate and private key using openssl
$ openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:2048 -keyout private.key -out certificate_pub.crt

  Generating Generating a 2048 bit RSA private key
  ..........................................................................................+++
  .............................+++
  writing new private key to 'private.key'
  -----
```

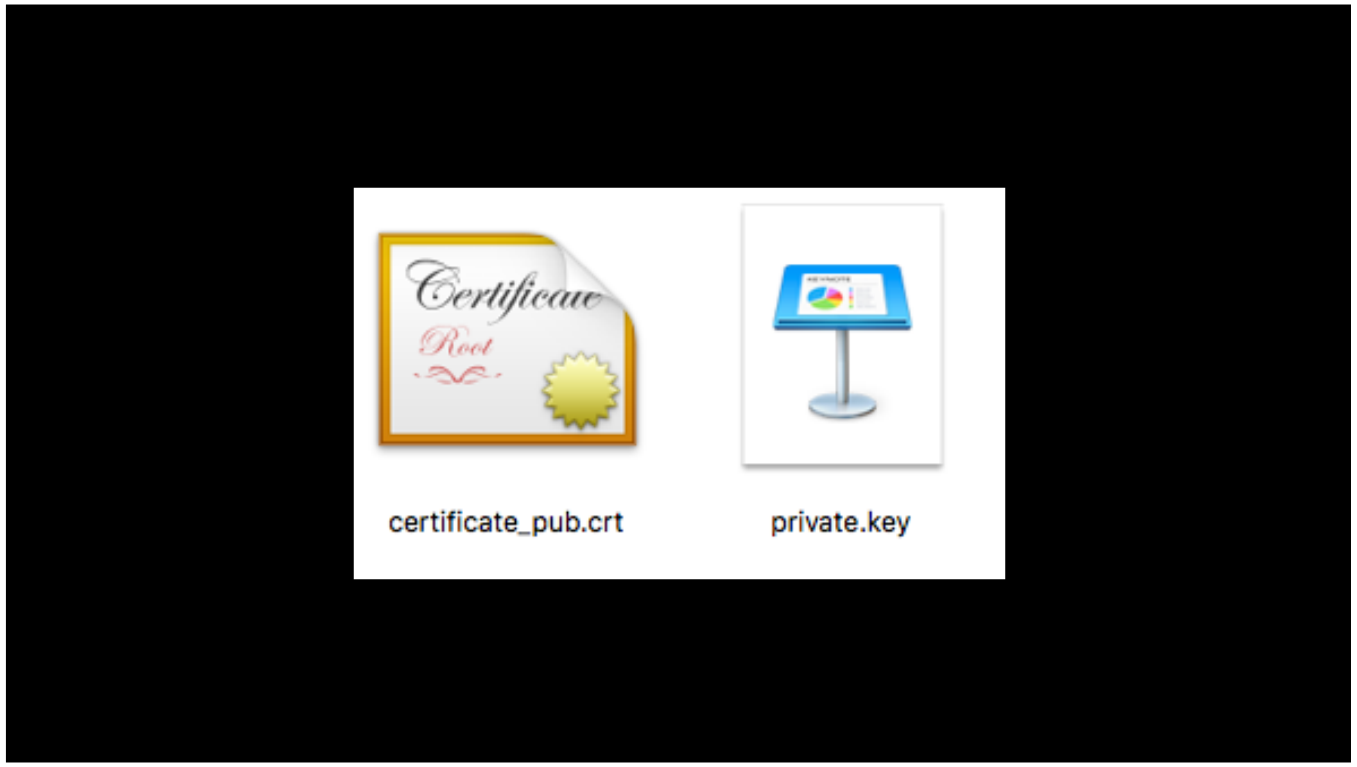
1095?

You need a certificate with a private and public key so your UST can talk securely to Adobe's UMAPI. That's straightforward to generate - the documentation has more details on how to do ti with openssl - you can do it in macOS with Terminal, or even Windows with Cygwin. Watch out though because the command it gives you will generate a cert that's only valid for one year even though the documentation itself recommends 3 years. If the certificate expires, the UST will stop performing actions to your identities in the Admin Console until you renew it.



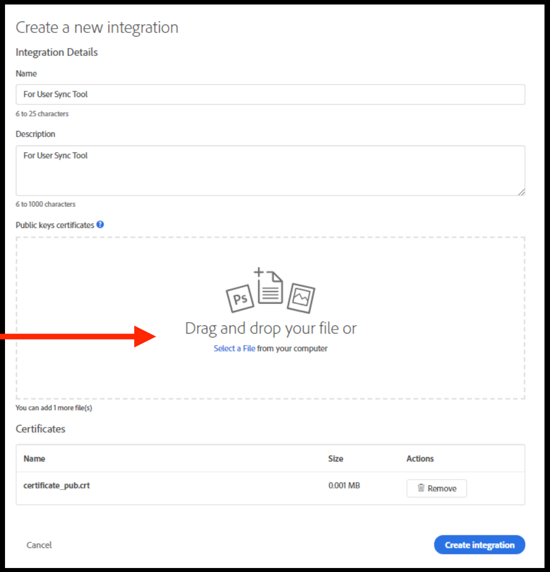
```
$ openssl req -x509 -sha256 -nodes -days 365 -newkey rsa:2048 -keyout private.key -out
certificate_pub.crt
Generating a 2048 bit RSA private key
...............................................................+++
............................+++
writing new private key to 'private.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:GB
State or Province Name (full name) []:Greater London
Locality Name (eg, city) []:London
Organization Name (eg, company) []:University of East London
Organizational Unit Name (eg, section) []:IT Services
Common Name (eg, fully qualified host name) []:Adobe IO Signing Certificate
Email Address []:itlicensing@uel.ac.uk
```

If you've never generated one of these before, be aware that openssl asks you a few questions. I don't think it matters to the UST what you put here but I'd match whatever your organisation normally does for certs.



You'll end up with a couple of files - the one on your left is your public certificate, the one on your right is your private key - keep that one really safe!

- https://www.adobe.io/
- Create a new integration (Service Account)
  - —> Access an API
  - —> User Management API



Create a new integration

Integration Details

Name

For User Sync Tool

6 to 25 characters

Description

For User Sync Tool

6 to 1000 characters

Public keys certificates

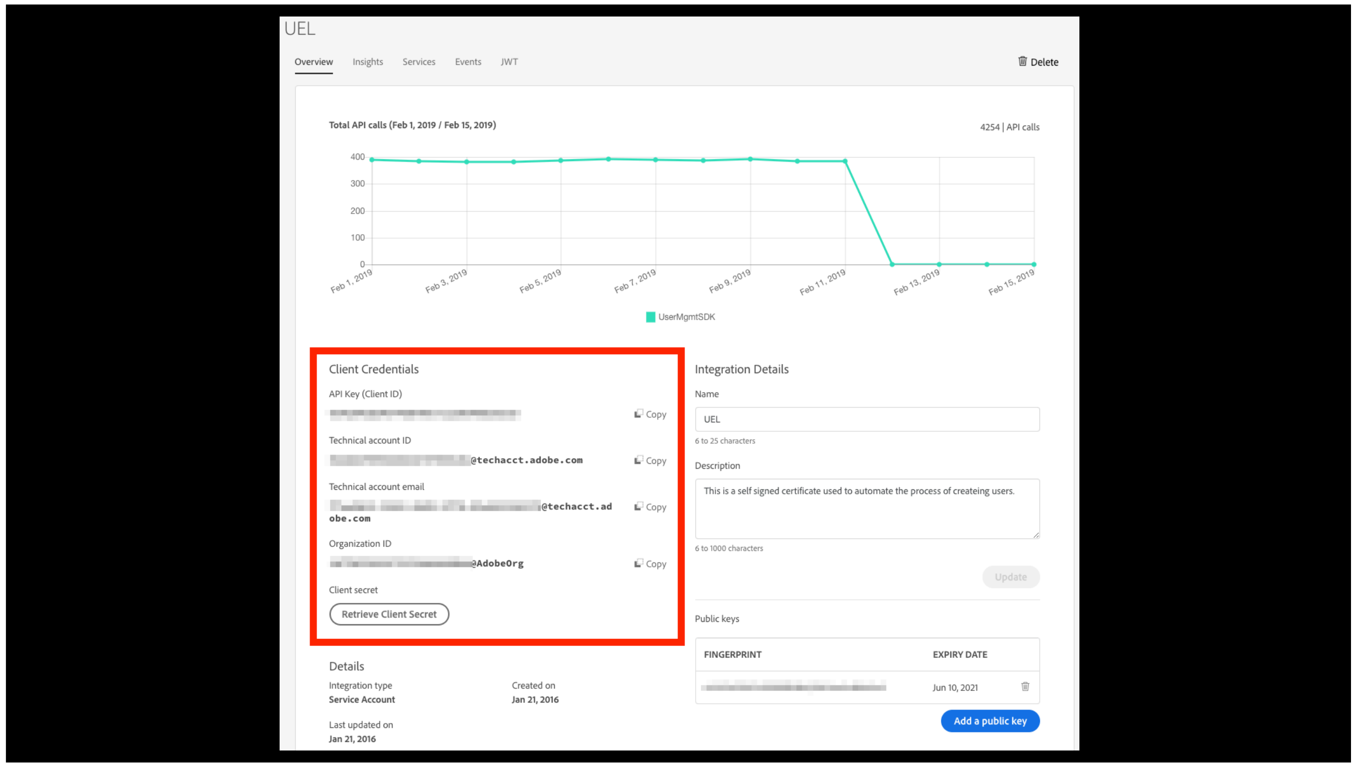
Drag and drop your file or
Select a File from your computer

You can add 1 more file(s)

Certificates

Name	Size	Actions
certificate_pub.crt	0.001 MB	Remove

The next thing to do is create a new Adobe IO integration, known as a Service Account. The UST connect to this when it performs actions on identities in your Admin Console. You'll sign in to www.adobe.io with the an Adobe ID that has system administrator access in your Admin Console, then follow the steps to create a new integration to access an API, specifically the User Management API. Give your integration a name and import the public certificate you just generated.



Total API calls (Feb 1, 2019 / Feb 15, 2019)

4254 | API calls

Client Credentials

API Key (Client ID)
[redacted] Copy

Technical account ID
`[redacted]@techacct.adobe.com` Copy

Technical account email
`[redacted]@techacct.adobe.com` Copy

Organization ID
`[redacted]@AdobeOrg` Copy

Client secret
Retrieve Client Secret

Details

Integration type	Created on
Service Account	**Jan 21, 2016**
Last updated on	
Jan 21, 2016	

Integration Details

Name
UEL
6 to 25 characters

Description
This is a self signed certificate used to automate the process of createing users.
6 to 1000 characters

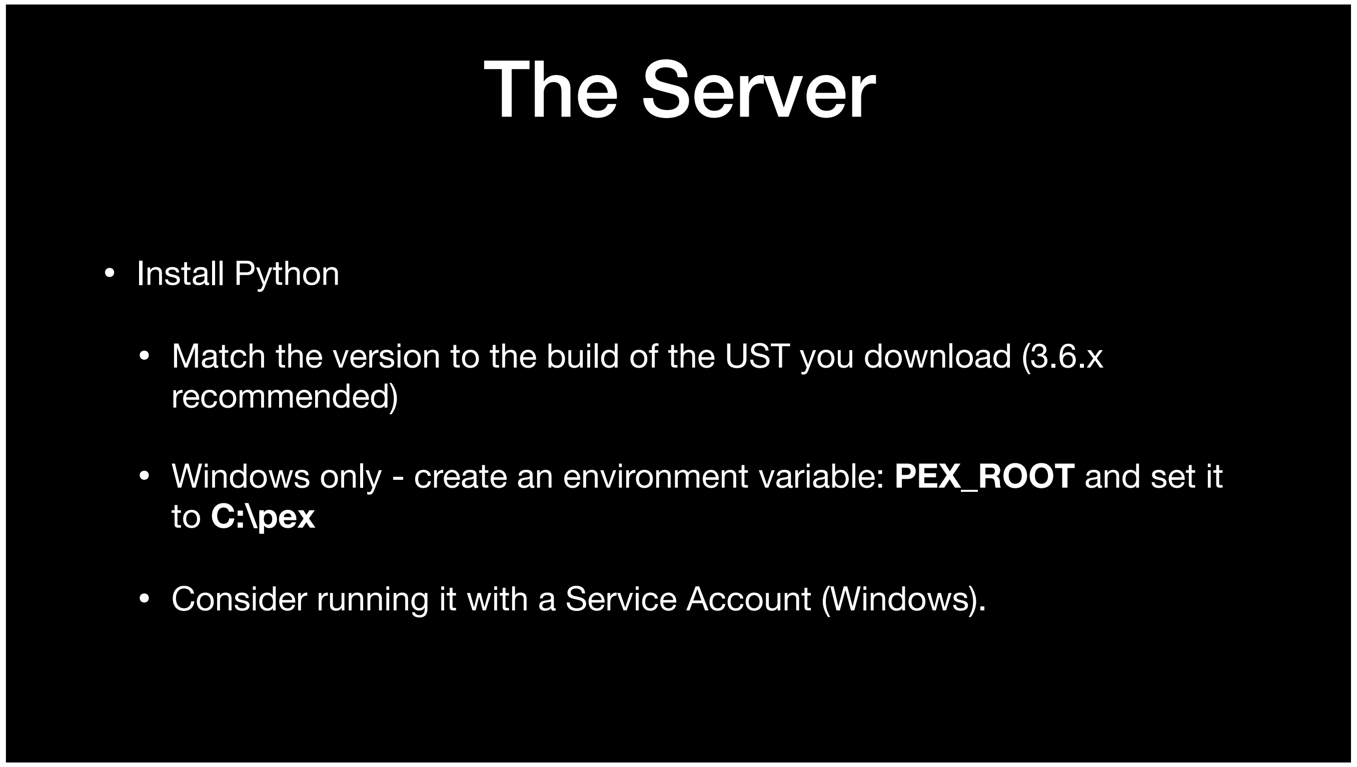
Update

Public keys

FINGERPRINT	EXPIRY DATE
[redacted]	Jun 10, 2021

Add a public key

When you view your integration, you'll need the details under Client Credentials later…



The Server

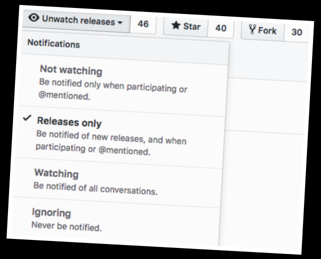
- Install Python
  - Match the version to the build of the UST you download (3.6.x recommended)
  - Windows only - create an environment variable: **PEX_ROOT** and set it to **C:\pex**
  - Consider running it with a Service Account (Windows).

You can run the UST on any computer but Adobe do recommend you put it on a server. It's not terribly demanding or resource-heavy, so I decided to install it on a couple of servers we also use for network licensing some of our software. I did this for redundancy - so it's running in 2 of our physical locations.

I don't know about other platforms but for Windows, I have the UST run as a scheduled task under a service account. This is so I can store various secure credentials it needs to access in the Windows Credentials Manager for that account. It's similar to the Keychain in macOS.

Get the UST!

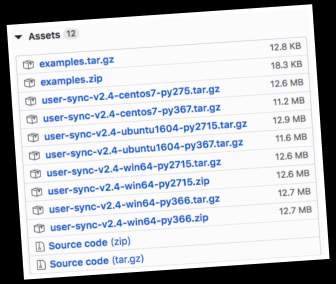
- https://github.com/adobe-apiplatform/user-sync.py/releases
- Watch for new releases
- Grab the release that best suits your environment
    - `user-sync-vx.x-platform-pythonversion`
- Grab **examples** too!



Unwatch releases 46 | Star 40 | Fork 30

Notifications

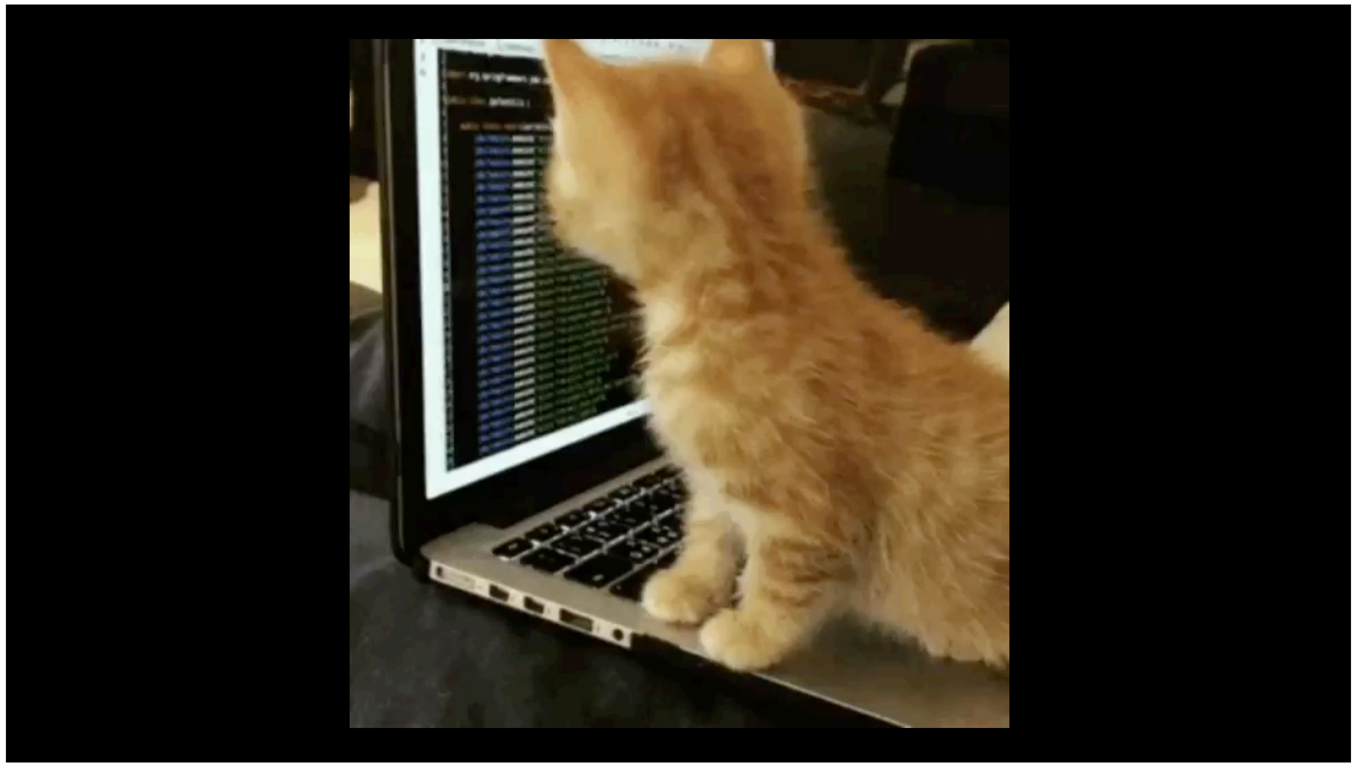
- **Not watching**
  Be notified only when participating or @mentioned.
- ✓ **Releases only**
  Be notified of new releases, and when participating or @mentioned.
- **Watching**
  Be notified of all conversations.
- **Ignoring**
  Never be notified.



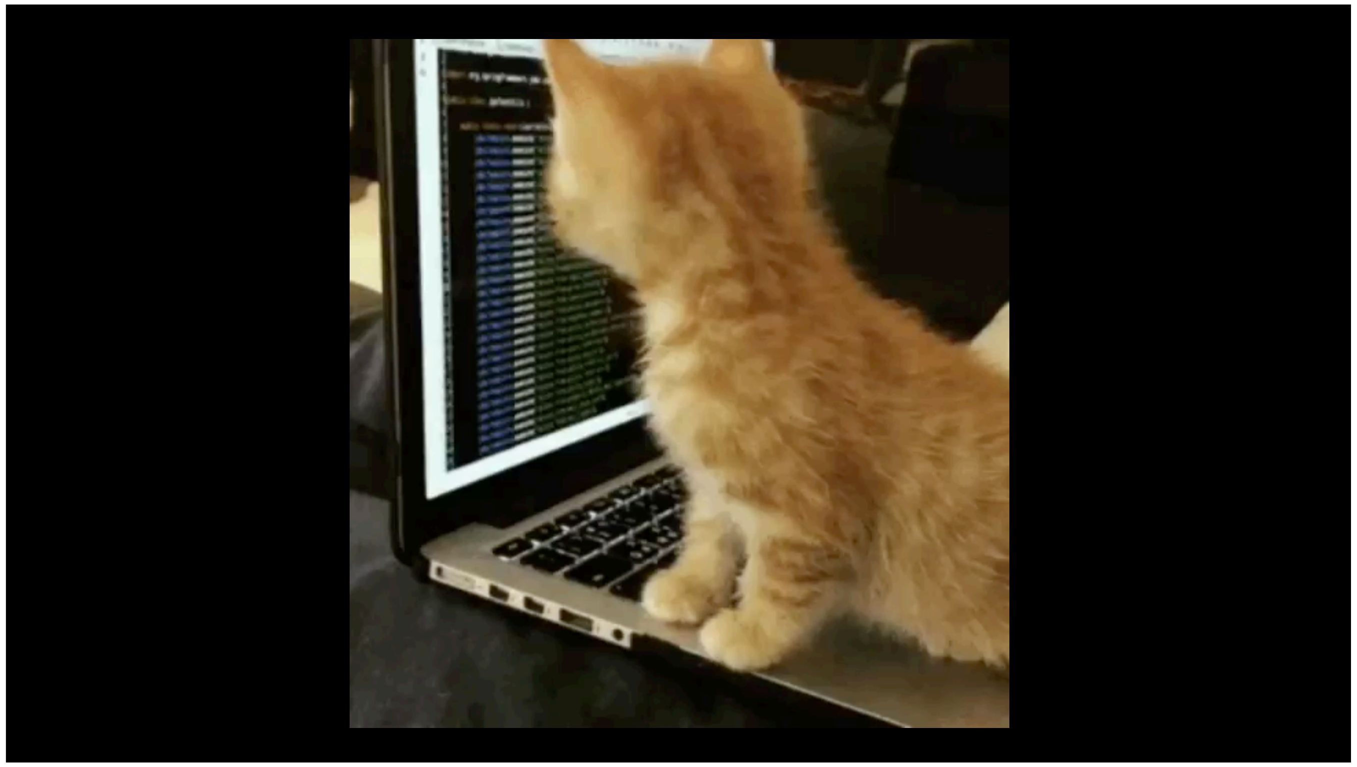
Assets 12

examples.tar.gz	12.8 KB
examples.zip	18.3 KB
user-sync-v2.4-centos7-py275.tar.gz	12.6 MB
user-sync-v2.4-centos7-py367.tar.gz	11.2 MB
user-sync-v2.4-ubuntu1604-py2715.tar.gz	12.9 MB
user-sync-v2.4-ubuntu1604-py367.tar.gz	11.6 MB
user-sync-v2.4-win64-py2715.tar.gz	12.6 MB
user-sync-v2.4-win64-py2715.zip	12.6 MB
user-sync-v2.4-win64-py366.tar.gz	12.7 MB
user-sync-v2.4-win64-py366.zip	12.7 MB
Source code (zip)	
Source code (tar.gz)	

You can grab the UST itself straight the GitHub repository. I’d recommend that you watch the repo for new releases. And make sure you download the right version for your environment - which is basically your target OS and Python version. You’ll also need the examples archive as well.



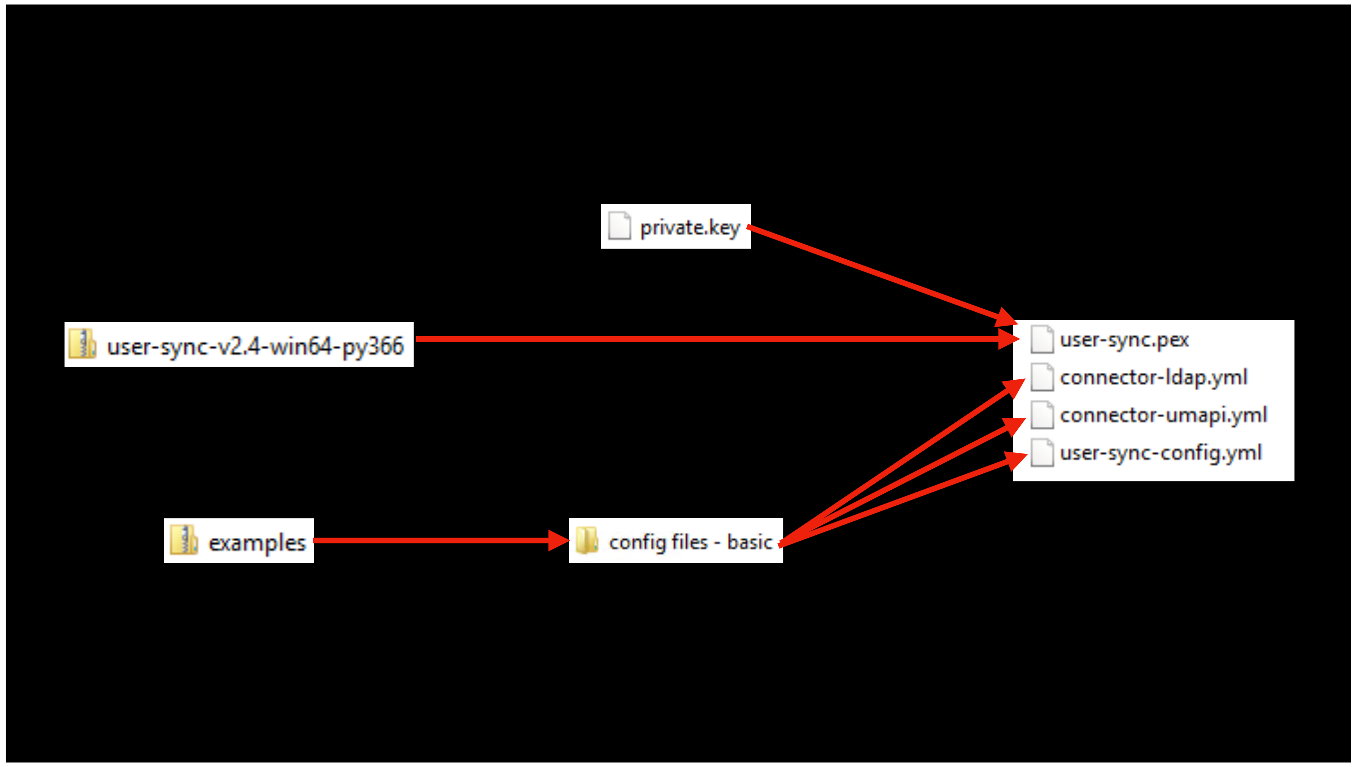
And now… we’re ready to set it up…



And now… we’re ready to set it up…

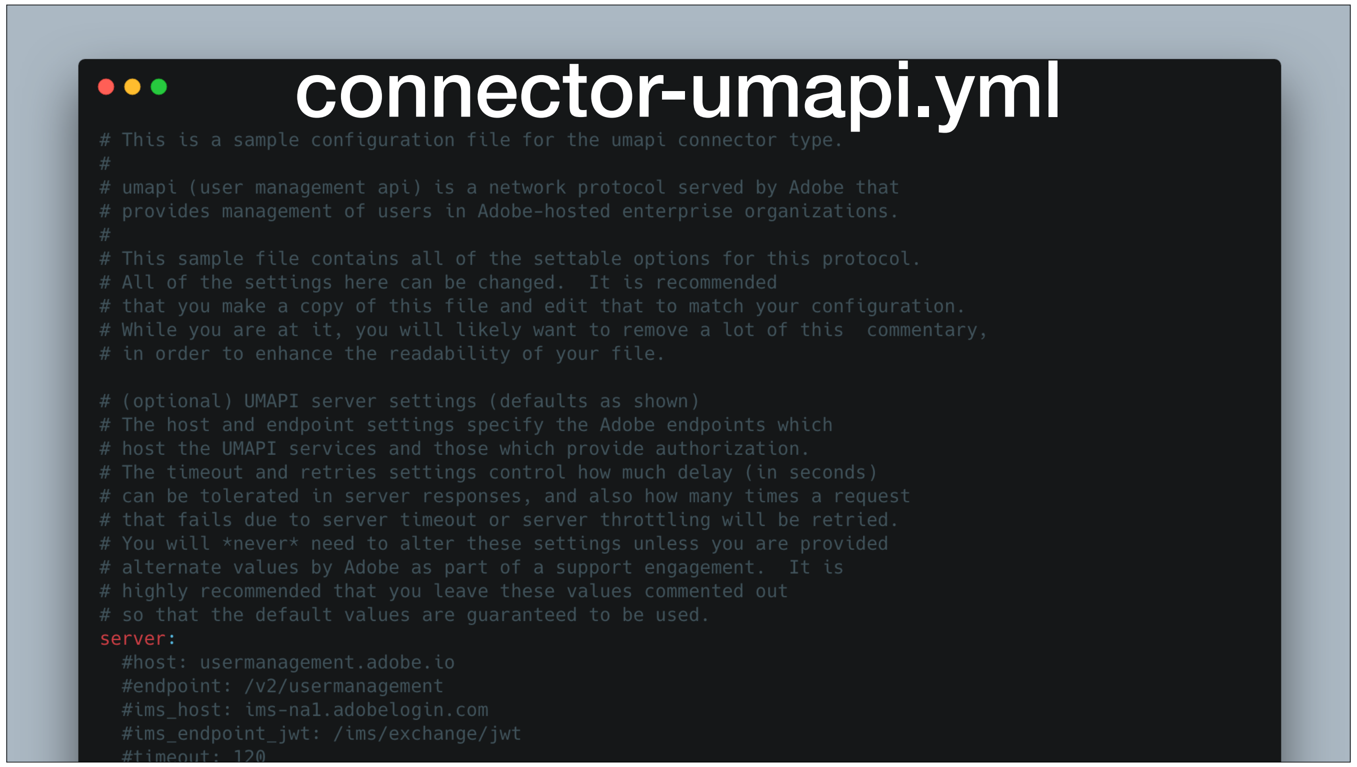
Setup

Now, how you do this well depend on your environment. For me, I’m pointing the UST at an Active Directory based server, using its LDAP connector, so a lot of what I’ll cover will be specific to that, but hopefully you’ll be able to get the gist behind what I’m doing so you can do this in your environment.



So - you'll want to create a folder structure of sorts on your server and extract all the bits you need to it.

In my case, that's the user-sync python script and three of the configuration files in the examples archive. Specifically the two that configure the LDAP and UMAPI connectors, plus the one that configures the UST itself. Also create a folder for the private key from the certificate you generated earlier.



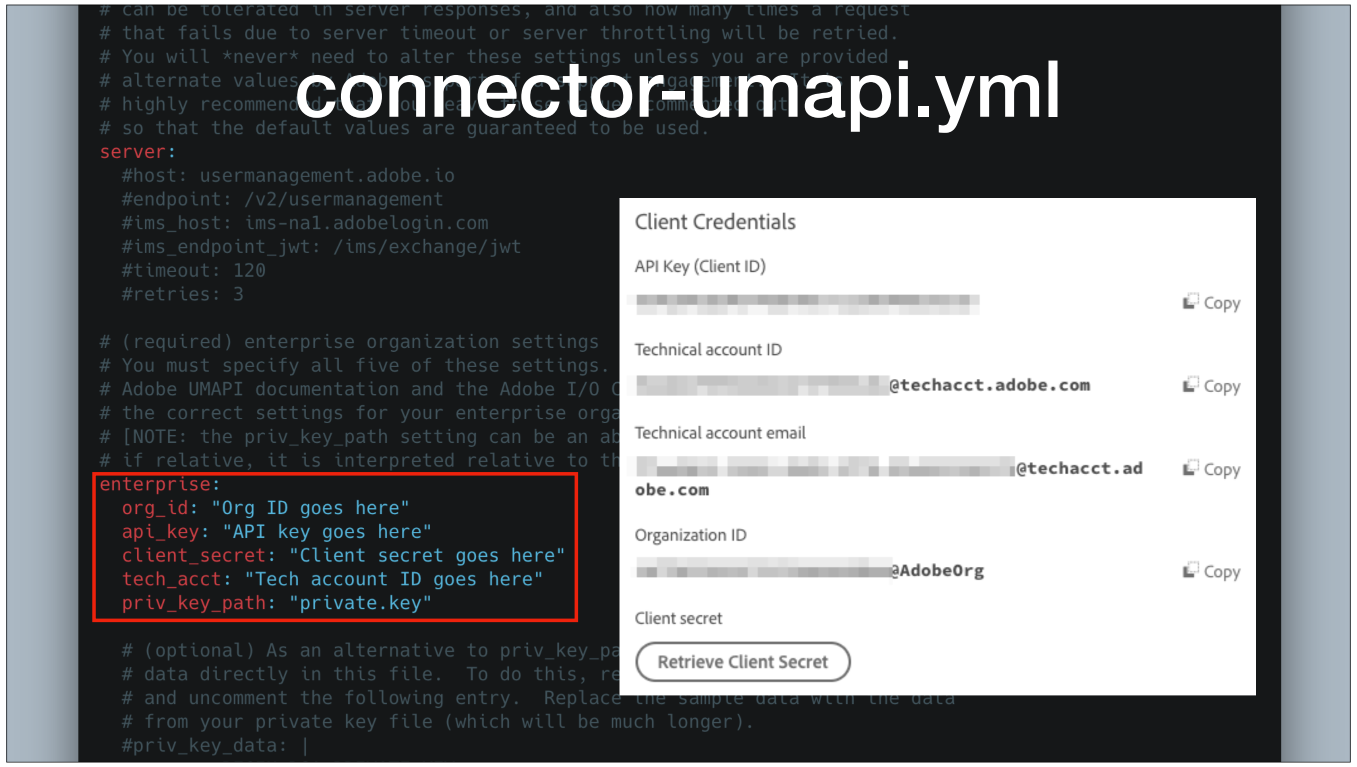
connector-umapi.yml

```yaml
# This is a sample configuration file for the umapi connector type.
#
# umapi (user management api) is a network protocol served by Adobe that
# provides management of users in Adobe-hosted enterprise organizations.
#
# This sample file contains all of the settable options for this protocol.
# All of the settings here can be changed.  It is recommended
# that you make a copy of this file and edit that to match your configuration.
# While you are at it, you will likely want to remove a lot of this  commentary,
# in order to enhance the readability of your file.

# (optional) UMAPI server settings (defaults as shown)
# The host and endpoint settings specify the Adobe endpoints which
# host the UMAPI services and those which provide authorization.
# The timeout and retries settings control how much delay (in seconds)
# can be tolerated in server responses, and also how many times a request
# that fails due to server timeout or server throttling will be retried.
# You will *never* need to alter these settings unless you are provided
# alternate values by Adobe as part of a support engagement.  It is
# highly recommended that you leave these values commented out
# so that the default values are guaranteed to be used.
server:
  #host: usermanagement.adobe.io
  #endpoint: /v2/usermanagement
  #ims_host: ims-na1.adobelogin.com
  #ims_endpoint_jwt: /ims/exchange/jwt
  #timeout: 120
```

Now let's look at those three YAML files. YML stands for Yet Another Markup Language. What's good about these is the comments - they're really well documented, so do read through those. The examples I'll show are the unmodified versions you get from the archive.

Let's start with the UMAPI connector…

connector-umapi.yml



```
# can be tolerated in server responses, and also how many times a request
# that fails due to server timeout or server throttling will be retried.
# You will *never* need to alter these settings unless you are provided
# alternate values
# highly recommend
# so that the default values are guaranteed to be used.
server:
  #host: usermanagement.adobe.io
  #endpoint: /v2/usermanagement
  #ims_host: ims-na1.adobelogin.com
  #ims_endpoint_jwt: /ims/exchange/jwt
  #timeout: 120
  #retries: 3

# (required) enterprise organization settings
# You must specify all five of these settings.
# Adobe UMAPI documentation and the Adobe I/O C
# the correct settings for your enterprise orga
# [NOTE: the priv_key_path setting can be an ab
# if relative, it is interpreted relative to th
enterprise:
  org_id: "Org ID goes here"
  api_key: "API key goes here"
  client_secret: "Client secret goes here"
  tech_acct: "Tech account ID goes here"
  priv_key_path: "private.key"

  # (optional) As an alternative to priv_key_pa
  # data directly in this file.  To do this, re
  # and uncomment the following entry.  Replace the sample data with the data
  # from your private key file (which will be much longer).
  #priv_key_data: |
```

Client Credentials

API Key (Client ID)

[redacted]

Technical account ID

[redacted]@techacct.adobe.com

Technical account email

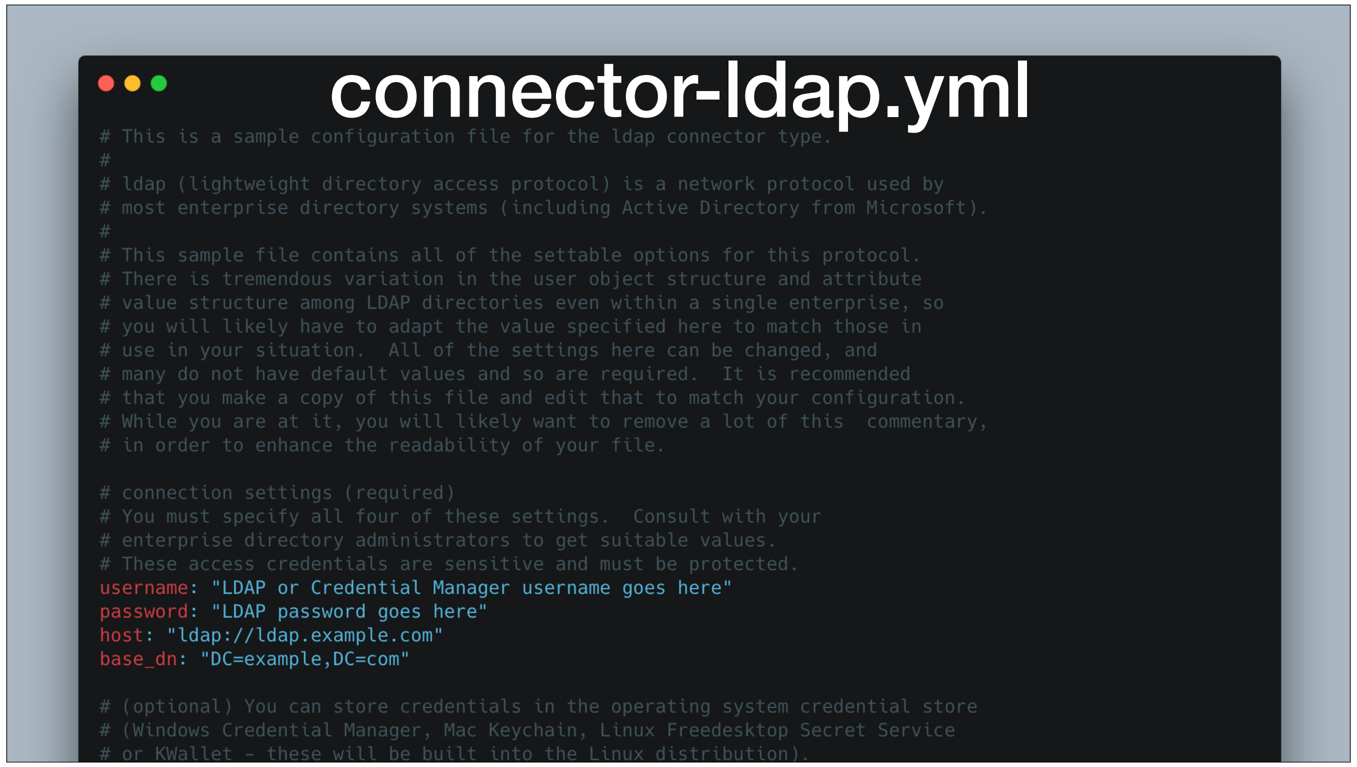
[redacted]@techacct.adobe.com

Organization ID

[redacted]@AdobeOrg

Client secret

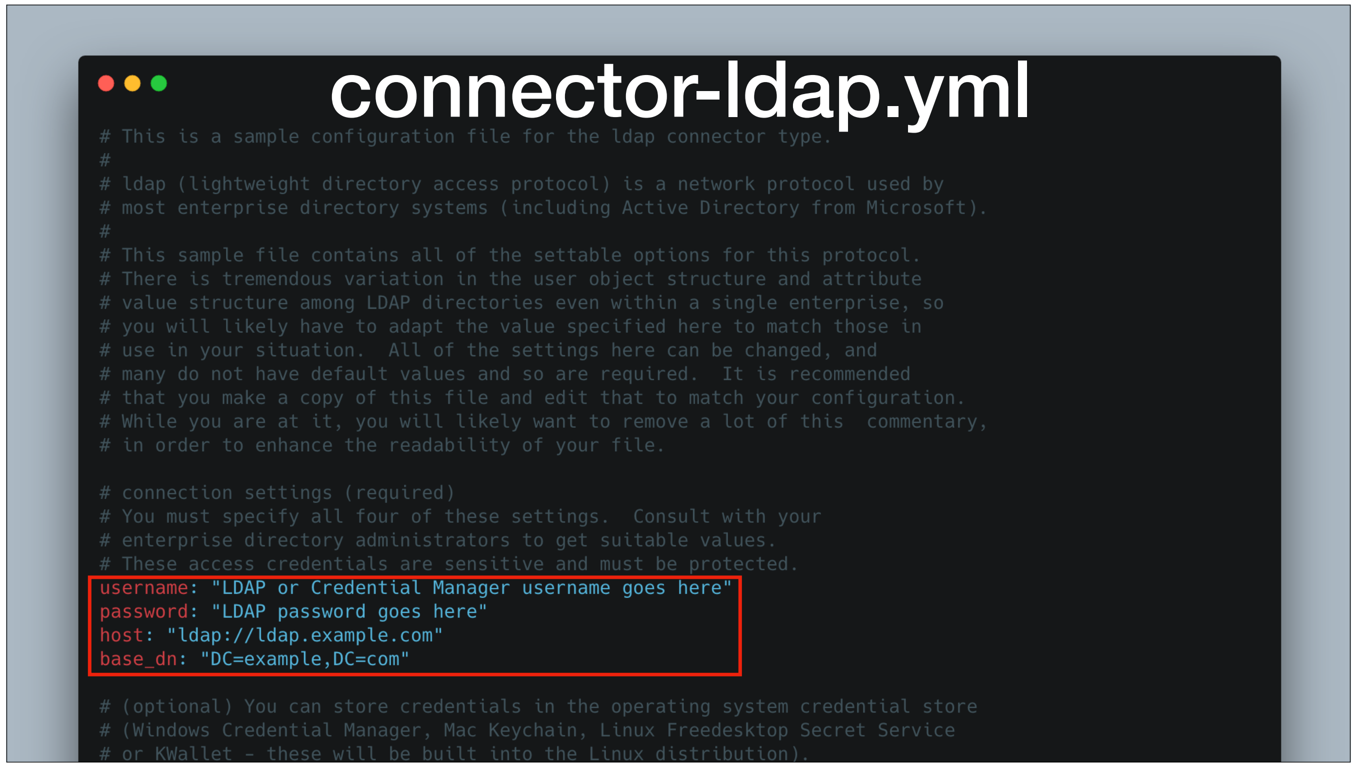
We need to tell it about the Adobe IO integration we made earlier. Specifically, the credentials for it and the filename for your private key. Log into adobe.io and open up your integration to get the credentials.



connector-ldap.yml

```
# This is a sample configuration file for the ldap connector type.
#
# ldap (lightweight directory access protocol) is a network protocol used by
# most enterprise directory systems (including Active Directory from Microsoft).
#
# This sample file contains all of the settable options for this protocol.
# There is tremendous variation in the user object structure and attribute
# value structure among LDAP directories even within a single enterprise, so
# you will likely have to adapt the value specified here to match those in
# use in your situation.  All of the settings here can be changed, and
# many do not have default values and so are required.  It is recommended
# that you make a copy of this file and edit that to match your configuration.
# While you are at it, you will likely want to remove a lot of this  commentary,
# in order to enhance the readability of your file.

# connection settings (required)
# You must specify all four of these settings.  Consult with your
# enterprise directory administrators to get suitable values.
# These access credentials are sensitive and must be protected.
username: "LDAP or Credential Manager username goes here"
password: "LDAP password goes here"
host: "ldap://ldap.example.com"
base_dn: "DC=example,DC=com"

# (optional) You can store credentials in the operating system credential store
# (Windows Credential Manager, Mac Keychain, Linux Freedesktop Secret Service
# or KWallet - these will be built into the Linux distribution).
```

Next up, if your user accounts live in an AD or LDAP server, is the LDAP connector. This one is fun…

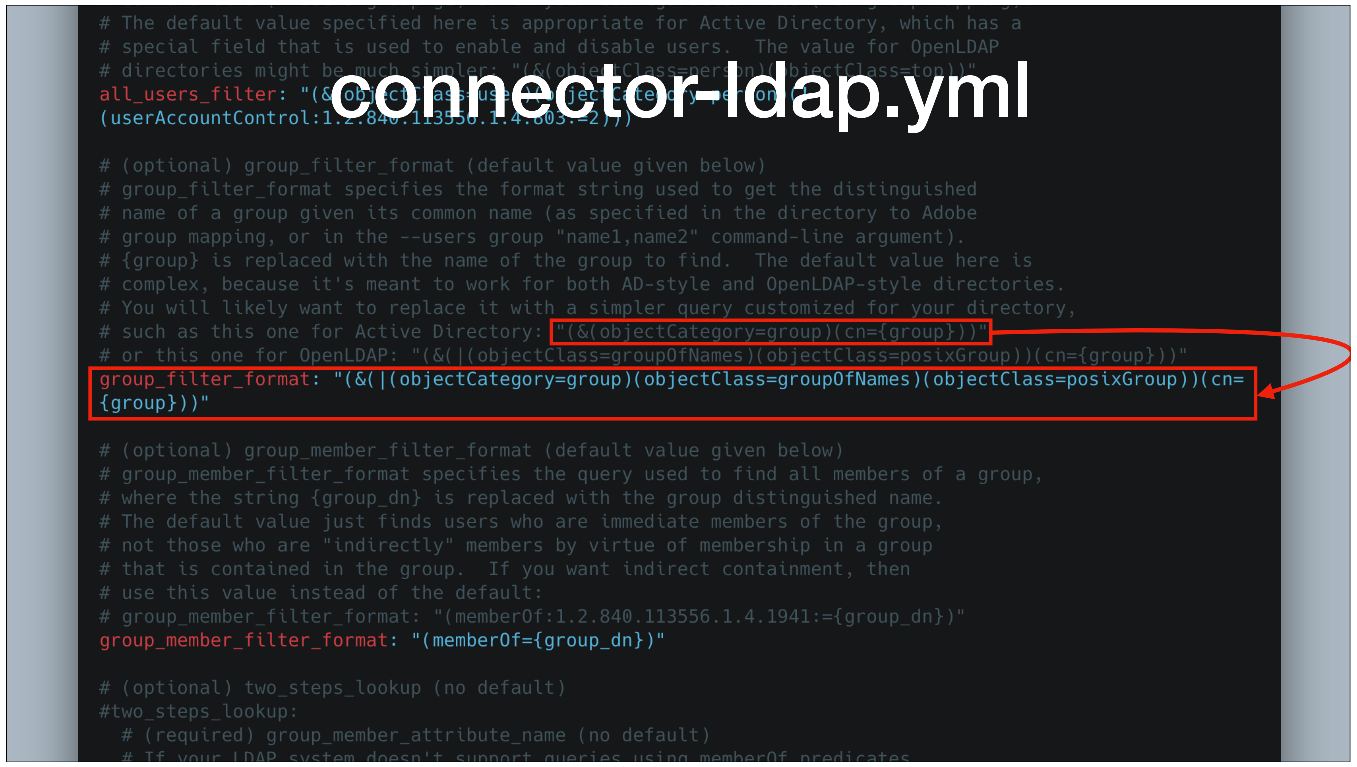


connector-ldap.yml

```
# This is a sample configuration file for the ldap connector type.
#
# ldap (lightweight directory access protocol) is a network protocol used by
# most enterprise directory systems (including Active Directory from Microsoft).
#
# This sample file contains all of the settable options for this protocol.
# There is tremendous variation in the user object structure and attribute
# value structure among LDAP directories even within a single enterprise, so
# you will likely have to adapt the value specified here to match those in
# use in your situation.  All of the settings here can be changed, and
# many do not have default values and so are required.  It is recommended
# that you make a copy of this file and edit that to match your configuration.
# While you are at it, you will likely want to remove a lot of this  commentary,
# in order to enhance the readability of your file.

# connection settings (required)
# You must specify all four of these settings.  Consult with your
# enterprise directory administrators to get suitable values.
# These access credentials are sensitive and must be protected.
username: "LDAP or Credential Manager username goes here"
password: "LDAP password goes here"
host: "ldap://ldap.example.com"
base_dn: "DC=example,DC=com"

# (optional) You can store credentials in the operating system credential store
# (Windows Credential Manager, Mac Keychain, Linux Freedesktop Secret Service
# or KWallet - these will be built into the Linux distribution).
```

Start with the details you need to connect to your directory service. These will be specific to your environment but the host does support an ldaps URL.

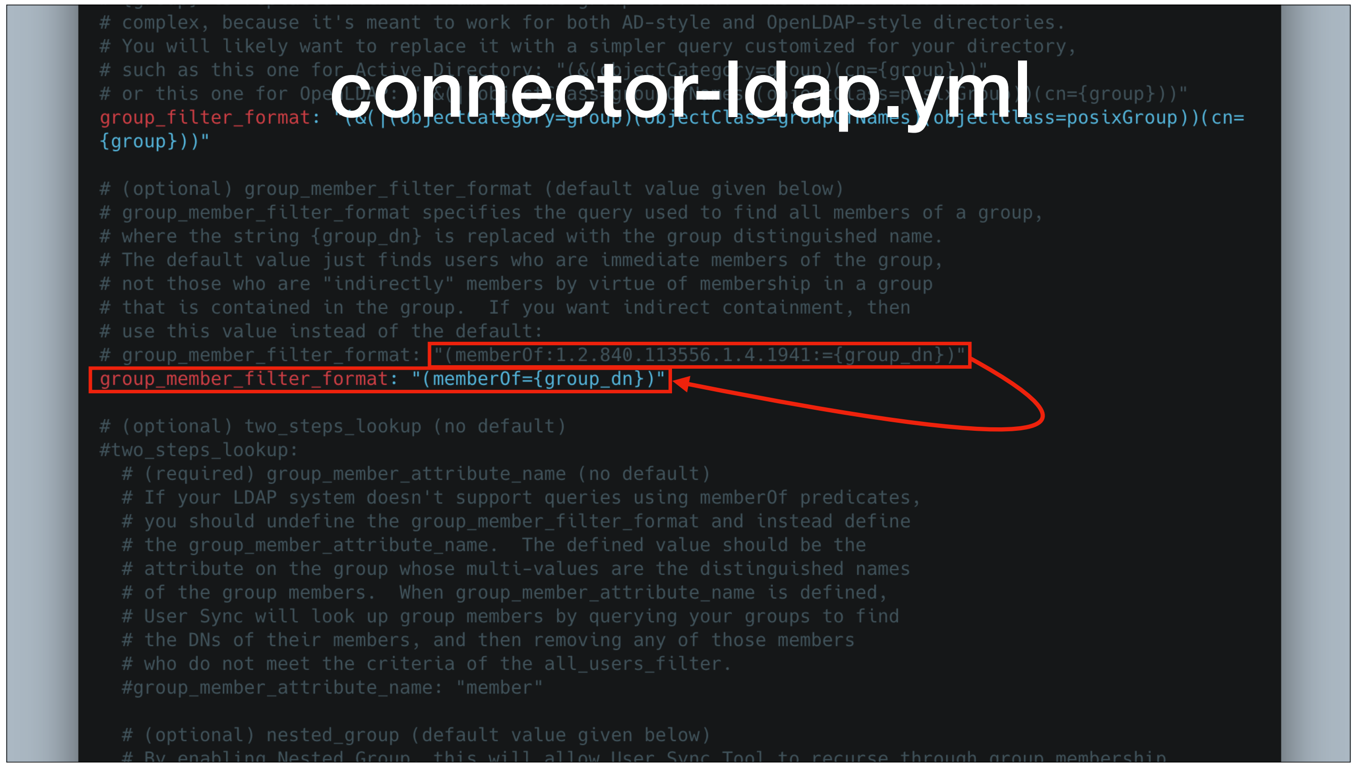
connector-ldap.yml



```
# The default value specified here is appropriate for Active Directory, which has a
# special field that is used to enable and disable users.  The value for OpenLDAP
# directories might be much simpler: "(&(objectClass=person)(objectClass=top))"
all_users_filter: "(&(objectClass=user)(objectCategory=person)(!
(userAccountControl:1.2.840.113556.1.4.803:=2)))"

# (optional) group_filter_format (default value given below)
# group_filter_format specifies the format string used to get the distinguished
# name of a group given its common name (as specified in the directory to Adobe
# group mapping, or in the --users group "name1,name2" command-line argument).
# {group} is replaced with the name of the group to find.  The default value here is
# complex, because it's meant to work for both AD-style and OpenLDAP-style directories.
# You will likely want to replace it with a simpler query customized for your directory,
# such as this one for Active Directory: "(&(objectCategory=group)(cn={group}))"
# or this one for OpenLDAP: "(&(|(objectClass=groupOfNames)(objectClass=posixGroup))(cn={group}))"
group_filter_format: "(&(|(objectCategory=group)(objectClass=groupOfNames)(objectClass=posixGroup))(cn=
{group}))"

# (optional) group_member_filter_format (default value given below)
# group_member_filter_format specifies the query used to find all members of a group,
# where the string {group_dn} is replaced with the group distinguished name.
# The default value just finds users who are immediate members of the group,
# not those who are "indirectly" members by virtue of membership in a group
# that is contained in the group.  If you want indirect containment, then
# use this value instead of the default:
# group_member_filter_format: "(memberOf:1.2.840.113556.1.4.1941:={group_dn})"
group_member_filter_format: "(memberOf={group_dn})"

# (optional) two_steps_lookup (no default)
#two_steps_lookup:
  # (required) group_member_attribute_name (no default)
  # If your LDAP system doesn't support queries using memberOf predicates
```

For Active Directory I could leave a lot of these settings at the default. But do read the comments above each setting. For the group_filter_format, I replaced it with a value that's more specific for AD

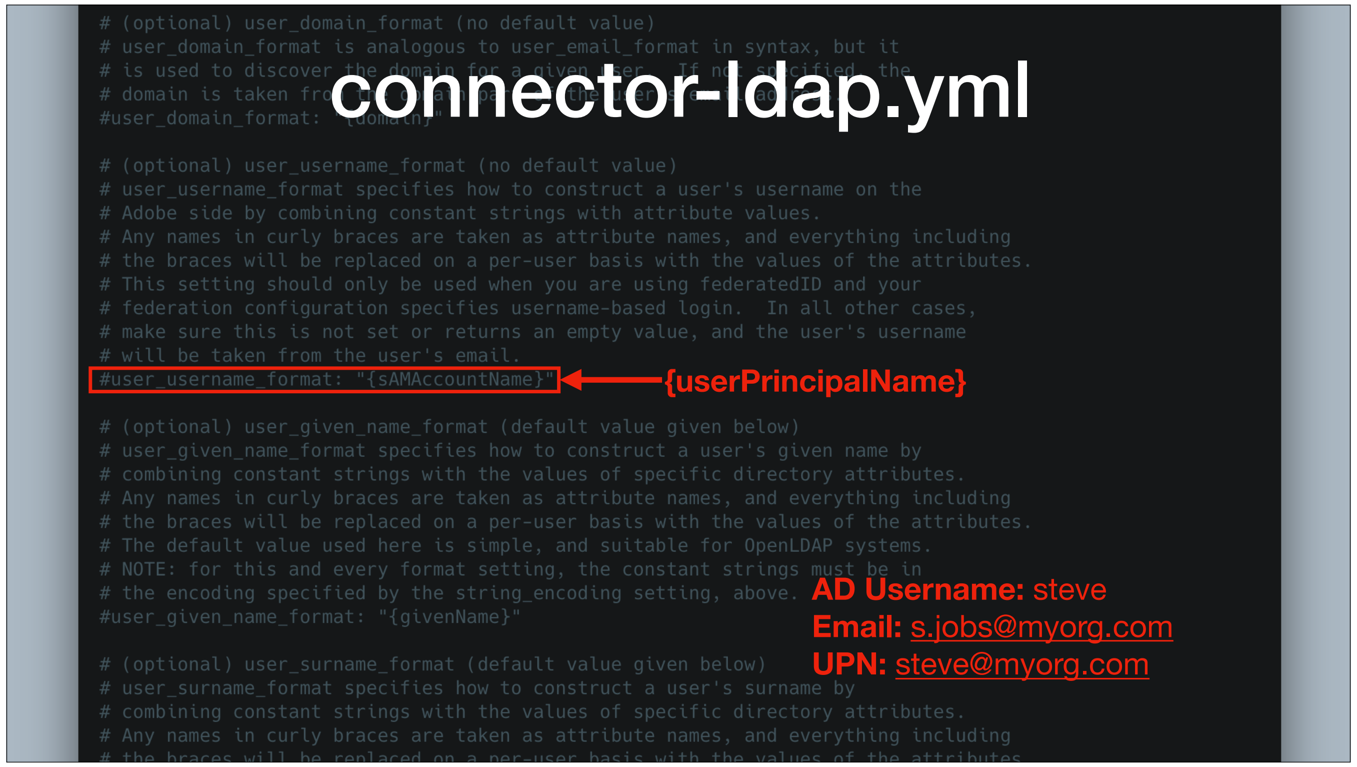
connector-ldap.yml



```
# complex, because it's meant to work for both AD-style and OpenLDAP-style directories.
# You will likely want to replace it with a simpler query customized for your directory,
# such as this one for Active Directory: "(&(objectCategory=group)(cn={group}))"
# or this one for OpenLDAP: "(&(|(objectClass=groupOfNames)(objectClass=posixGroup))(cn={group}))"
group_filter_format: "(&(|(objectCategory=group)(objectClass=groupOfNames)(objectclass=posixGroup))(cn=
{group}))"

# (optional) group_member_filter_format (default value given below)
# group_member_filter_format specifies the query used to find all members of a group,
# where the string {group_dn} is replaced with the group distinguished name.
# The default value just finds users who are immediate members of the group,
# not those who are "indirectly" members by virtue of membership in a group
# that is contained in the group.  If you want indirect containment, then
# use this value instead of the default:
# group_member_filter_format: "(memberOf:1.2.840.113556.1.4.1941:={group_dn})"
group_member_filter_format: "(memberOf={group_dn})"

# (optional) two_steps_lookup (no default)
#two_steps_lookup:
  # (required) group_member_attribute_name (no default)
  # If your LDAP system doesn't support queries using memberOf predicates,
  # you should undefine the group_member_filter_format and instead define
  # the group_member_attribute_name.  The defined value should be the
  # attribute on the group whose multi-values are the distinguished names
  # of the group members.  When group_member_attribute_name is defined,
  # User Sync will look up group members by querying your groups to find
  # the DNs of their members, and then removing any of those members
  # who do not meet the criteria of the all_users_filter.
  #group_member_attribute_name: "member"

  # (optional) nested_group (default value given below)
  # By enabling Nested Group, this will allow User Sync Tool to recurse through group membership
```

I also had to change the group_member_filter_format to the value suggested in the comments because the AD groups I'll be syncing don't contain the user objects; they actually contain nested groups for various departments or faculties, and the user objects are in those groups…

connector-ldap.yml



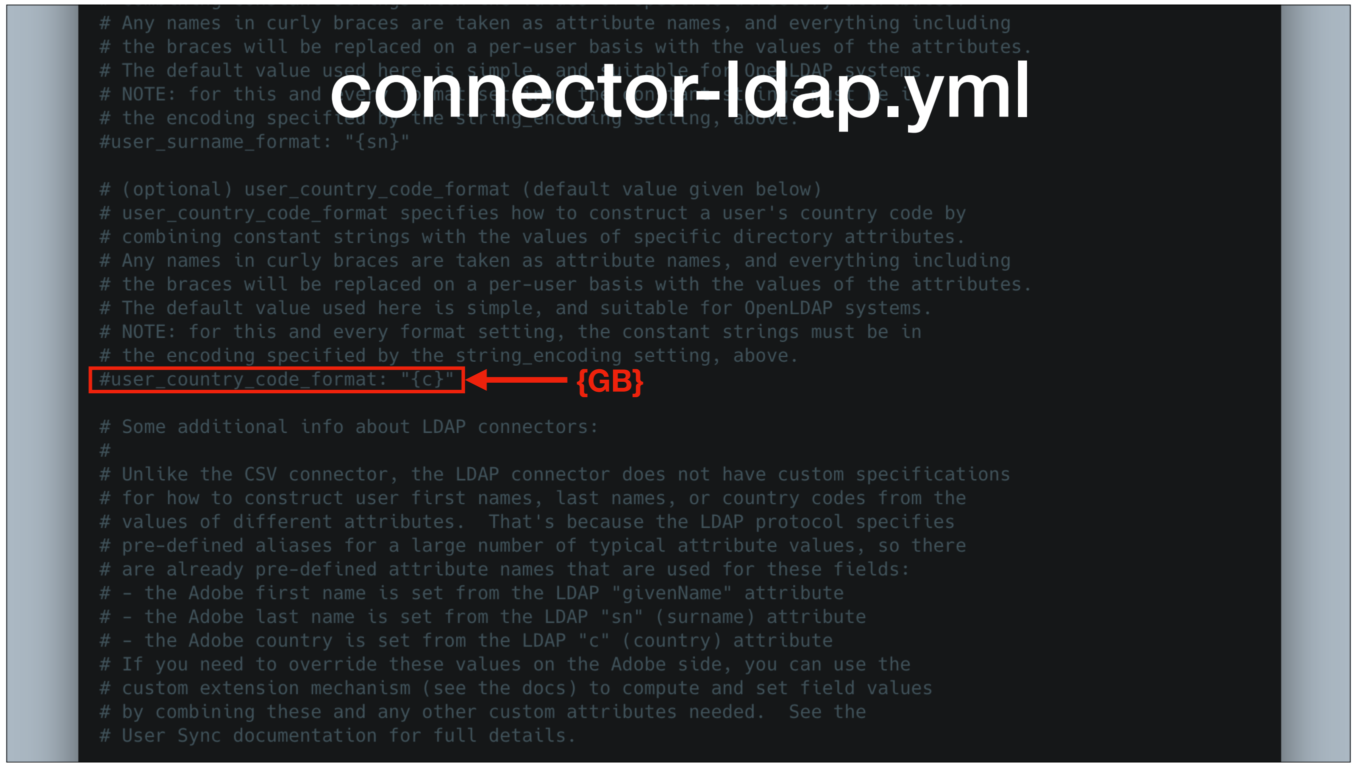
```
# (optional) user_domain_format (no default value)
# user_domain_format is analogous to user_email_format in syntax, but it
# is used to discover the domain for a given user.  If not specified, the
# domain is taken from the domain part of the user's email address.
#user_domain_format: "{domain}"

# (optional) user_username_format (no default value)
# user_username_format specifies how to construct a user's username on the
# Adobe side by combining constant strings with attribute values.
# Any names in curly braces are taken as attribute names, and everything including
# the braces will be replaced on a per-user basis with the values of the attributes.
# This setting should only be used when you are using federatedID and your
# federation configuration specifies username-based login.  In all other cases,
# make sure this is not set or returns an empty value, and the user's username
# will be taken from the user's email.
#user_username_format: "{sAMAccountName}"

# (optional) user_given_name_format (default value given below)
# user_given_name_format specifies how to construct a user's given name by
# combining constant strings with the values of specific directory attributes.
# Any names in curly braces are taken as attribute names, and everything including
# the braces will be replaced on a per-user basis with the values of the attributes.
# The default value used here is simple, and suitable for OpenLDAP systems.
# NOTE: for this and every format setting, the constant strings must be in
# the encoding specified by the string_encoding setting, above.
#user_given_name_format: "{givenName}"

# (optional) user_surname_format (default value given below)
# user_surname_format specifies how to construct a user's surname by
# combining constant strings with the values of specific directory attributes.
# Any names in curly braces are taken as attribute names, and everything including
# the braces will be replaced on a per-user basis with the values of the attributes.
```

{userPrincipalName}

AD Username: steve
Email: s.jobs@myorg.com
UPN: steve@myorg.com

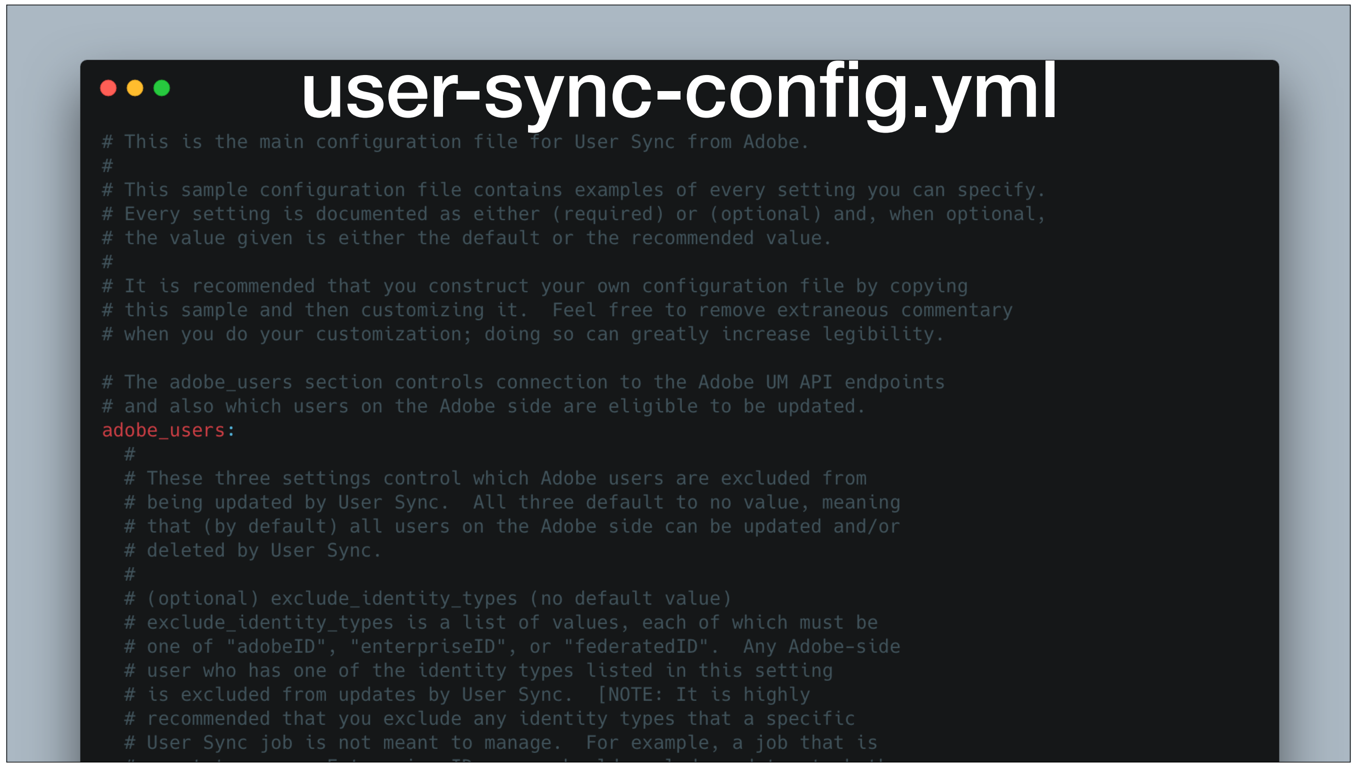
Identities in your Admin Console have an email address and username field. They default to using the user's email address for both. In our environment, they need to be different. We need the username to match the userPrincipalName attribute in the user's AD record because it's different from their email address. For us, the UPN is a combination of their AD username and our domain. Our SSO expects the UPN and will pass that to Adobe when users authenticate.

connector-ldap.yml



```
# Any names in curly braces are taken as attribute names, and everything including
# the braces will be replaced on a per-user basis with the values of the attributes.
# The default value used here is simple, and suitable for OpenLDAP systems.
# NOTE: for this and every format setting, the constant strings must be in
# the encoding specified by the string_encoding setting, above.
#user_surname_format: "{sn}"

# (optional) user_country_code_format (default value given below)
# user_country_code_format specifies how to construct a user's country code by
# combining constant strings with the values of specific directory attributes.
# Any names in curly braces are taken as attribute names, and everything including
# the braces will be replaced on a per-user basis with the values of the attributes.
# The default value used here is simple, and suitable for OpenLDAP systems.
# NOTE: for this and every format setting, the constant strings must be in
# the encoding specified by the string_encoding setting, above.
#user_country_code_format: "{c}"

# Some additional info about LDAP connectors:
#
# Unlike the CSV connector, the LDAP connector does not have custom specifications
# for how to construct user first names, last names, or country codes from the
# values of different attributes.  That's because the LDAP protocol specifies
# pre-defined aliases for a large number of typical attribute values, so there
# are already pre-defined attribute names that are used for these fields:
# - the Adobe first name is set from the LDAP "givenName" attribute
# - the Adobe last name is set from the LDAP "sn" (surname) attribute
# - the Adobe country is set from the LDAP "c" (country) attribute
# If you need to override these values on the Adobe side, you can use the
# custom extension mechanism (see the docs) to compute and set field values
# by combining these and any other custom attributes needed.  See the
# User Sync documentation for full details.
```

Finally, I had to set the user_country_code_format to GB because our AD doesn't have an attribute for it - and I saw lots of errors in the sync tool's logs without it set.

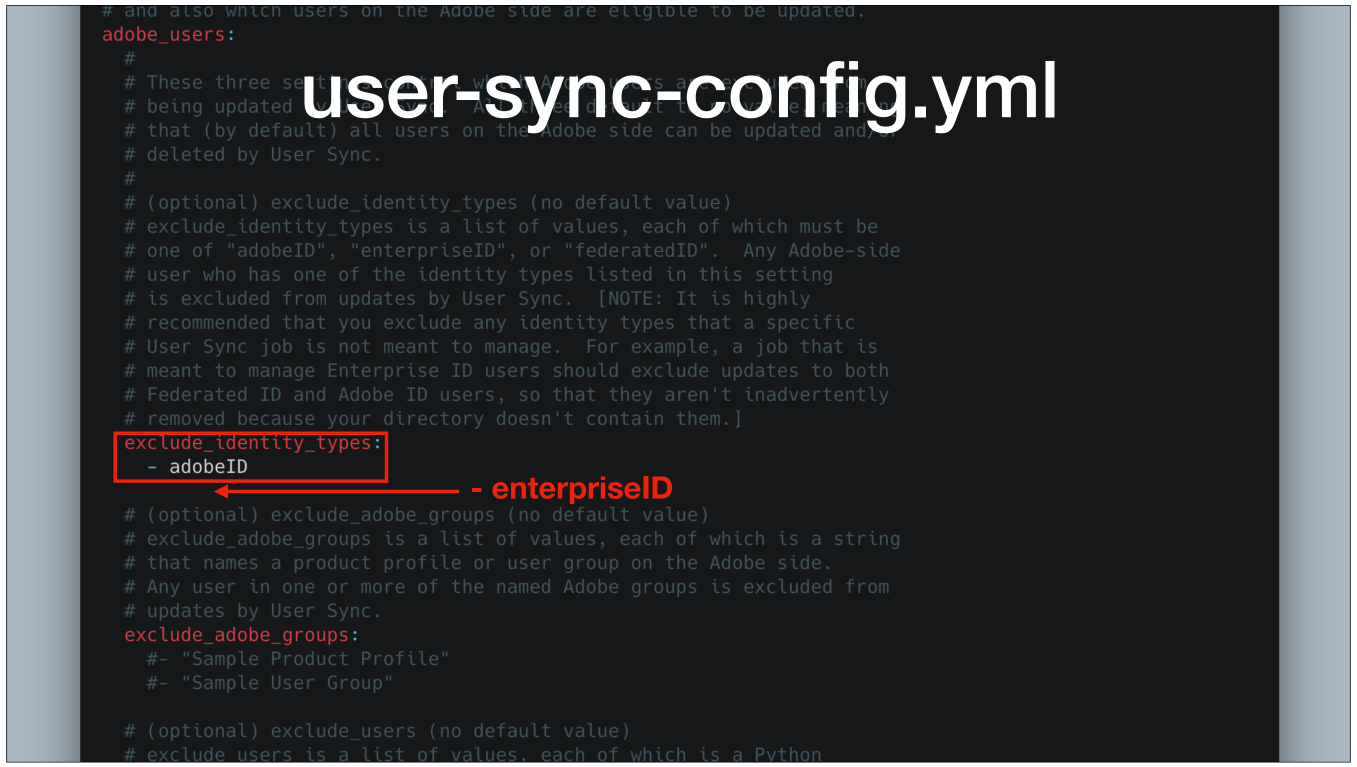


user-sync-config.yml

```
# This is the main configuration file for User Sync from Adobe.
#
# This sample configuration file contains examples of every setting you can specify.
# Every setting is documented as either (required) or (optional) and, when optional,
# the value given is either the default or the recommended value.
#
# It is recommended that you construct your own configuration file by copying
# this sample and then customizing it.  Feel free to remove extraneous commentary
# when you do your customization; doing so can greatly increase legibility.

# The adobe_users section controls connection to the Adobe UM API endpoints
# and also which users on the Adobe side are eligible to be updated.
adobe_users:
  #
  # These three settings control which Adobe users are excluded from
  # being updated by User Sync.  All three default to no value, meaning
  # that (by default) all users on the Adobe side can be updated and/or
  # deleted by User Sync.
  #
  # (optional) exclude_identity_types (no default value)
  # exclude_identity_types is a list of values, each of which must be
  # one of "adobeID", "enterpriseID", or "federatedID".  Any Adobe-side
  # user who has one of the identity types listed in this setting
  # is excluded from updates by User Sync.  [NOTE: It is highly
  # recommended that you exclude any identity types that a specific
  # User Sync job is not meant to manage.  For example, a job that is
```

Moving on to the last file - settings for the user sync tool itself.

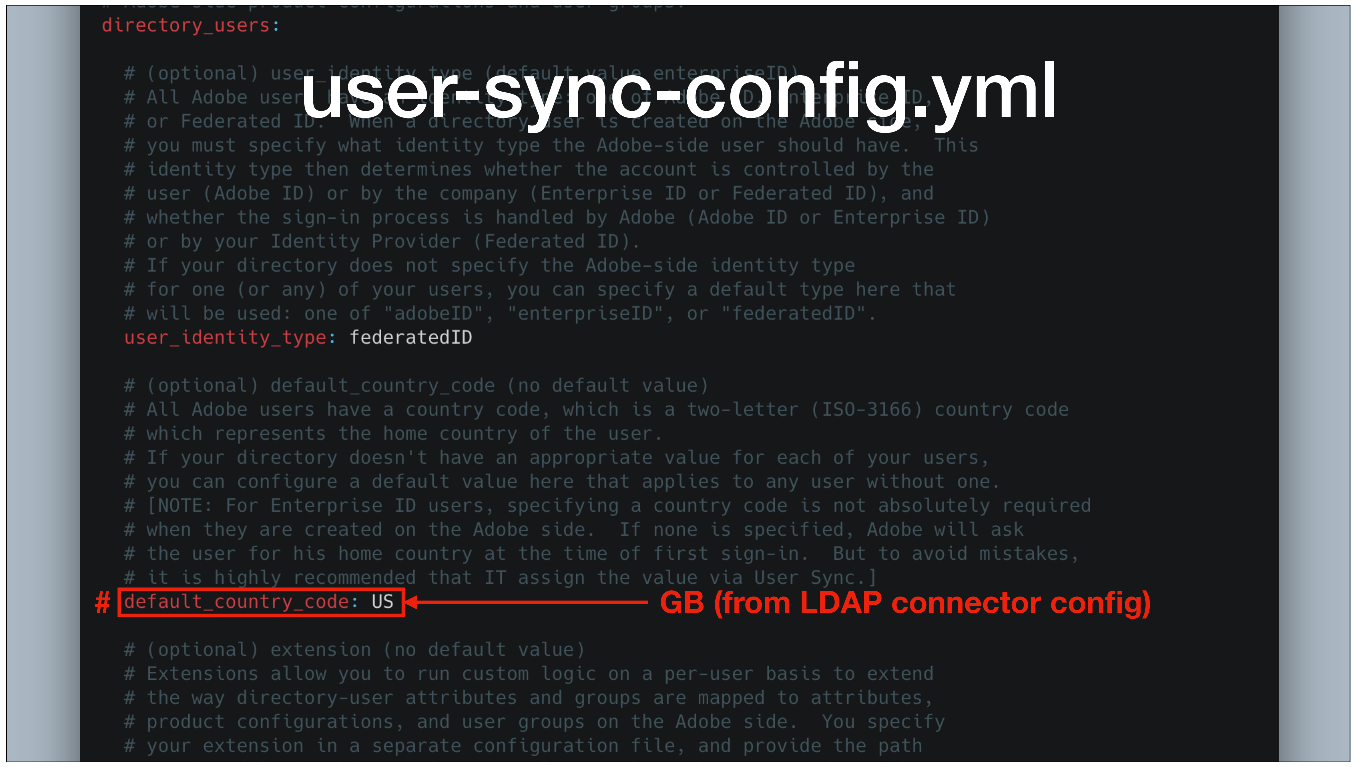
user-sync-config.yml



```
  # and also which users on the Adobe side are eligible to be updated.
adobe_users:
  #
  # These three settings control which Adobe users are excluded from
  # being updated by User Sync.  All three default to no value, meaning
  # that (by default) all users on the Adobe side can be updated and/or
  # deleted by User Sync.
  #
  # (optional) exclude_identity_types (no default value)
  # exclude_identity_types is a list of values, each of which must be
  # one of "adobeID", "enterpriseID", or "federatedID".  Any Adobe-side
  # user who has one of the identity types listed in this setting
  # is excluded from updates by User Sync.  [NOTE: It is highly
  # recommended that you exclude any identity types that a specific
  # User Sync job is not meant to manage.  For example, a job that is
  # meant to manage Enterprise ID users should exclude updates to both
  # Federated ID and Adobe ID users, so that they aren't inadvertently
  # removed because your directory doesn't contain them.]
  exclude_identity_types:
    - adobeID

  # (optional) exclude_adobe_groups (no default value)
  # exclude_adobe_groups is a list of values, each of which is a string
  # that names a product profile or user group on the Adobe side.
  # Any user in one or more of the named Adobe groups is excluded from
  # updates by User Sync.
  exclude_adobe_groups:
    #- "Sample Product Profile"
    #- "Sample User Group"

  # (optional) exclude_users (no default value)
  # exclude_users is a list of values, each of which is a Python
```

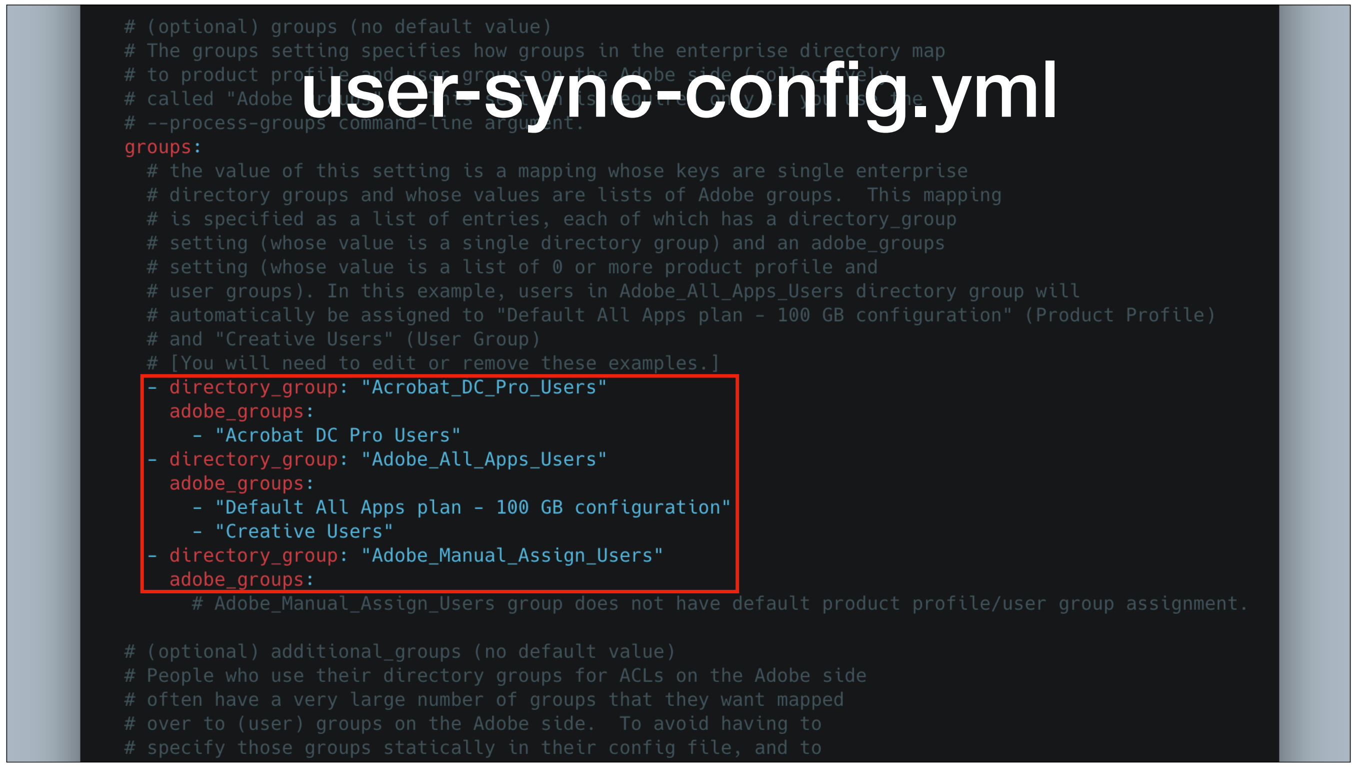
Under the adobe_users settings, you need to specify which identity types you're excluding - as in, ones the UST won't ever touch in your Admin Console. We're using Federated IDs so we don't want to manage Adobe IDs or Enterprise IDs in our console - I'll append enterpriseID to that list.

user-sync-config.yml



```
directory_users:

  # (optional) user_identity_type (default value enterpriseID)
  # All Adobe users have an identity type: one of Adobe ID, Enterprise ID,
  # or Federated ID.  When a directory user is created on the Adobe side,
  # you must specify what identity type the Adobe-side user should have.  This
  # identity type then determines whether the account is controlled by the
  # user (Adobe ID) or by the company (Enterprise ID or Federated ID), and
  # whether the sign-in process is handled by Adobe (Adobe ID or Enterprise ID)
  # or by your Identity Provider (Federated ID).
  # If your directory does not specify the Adobe-side identity type
  # for one (or any) of your users, you can specify a default type here that
  # will be used: one of "adobeID", "enterpriseID", or "federatedID".
  user_identity_type: federatedID

  # (optional) default_country_code (no default value)
  # All Adobe users have a country code, which is a two-letter (ISO-3166) country code
  # which represents the home country of the user.
  # If your directory doesn't have an appropriate value for each of your users,
  # you can configure a default value here that applies to any user without one.
  # [NOTE: For Enterprise ID users, specifying a country code is not absolutely required
  # when they are created on the Adobe side.  If none is specified, Adobe will ask
  # the user for his home country at the time of first sign-in.  But to avoid mistakes,
  # it is highly recommended that IT assign the value via User Sync.]
# default_country_code: US

  # (optional) extension (no default value)
  # Extensions allow you to run custom logic on a per-user basis to extend
  # the way directory-user attributes and groups are mapped to attributes,
  # product configurations, and user groups on the Adobe side.  You specify
  # your extension in a separate configuration file, and provide the path
```

Under the directory_users settings, I commented out the default_country_setting because I specified it in the LDAP connector config file to be GB.

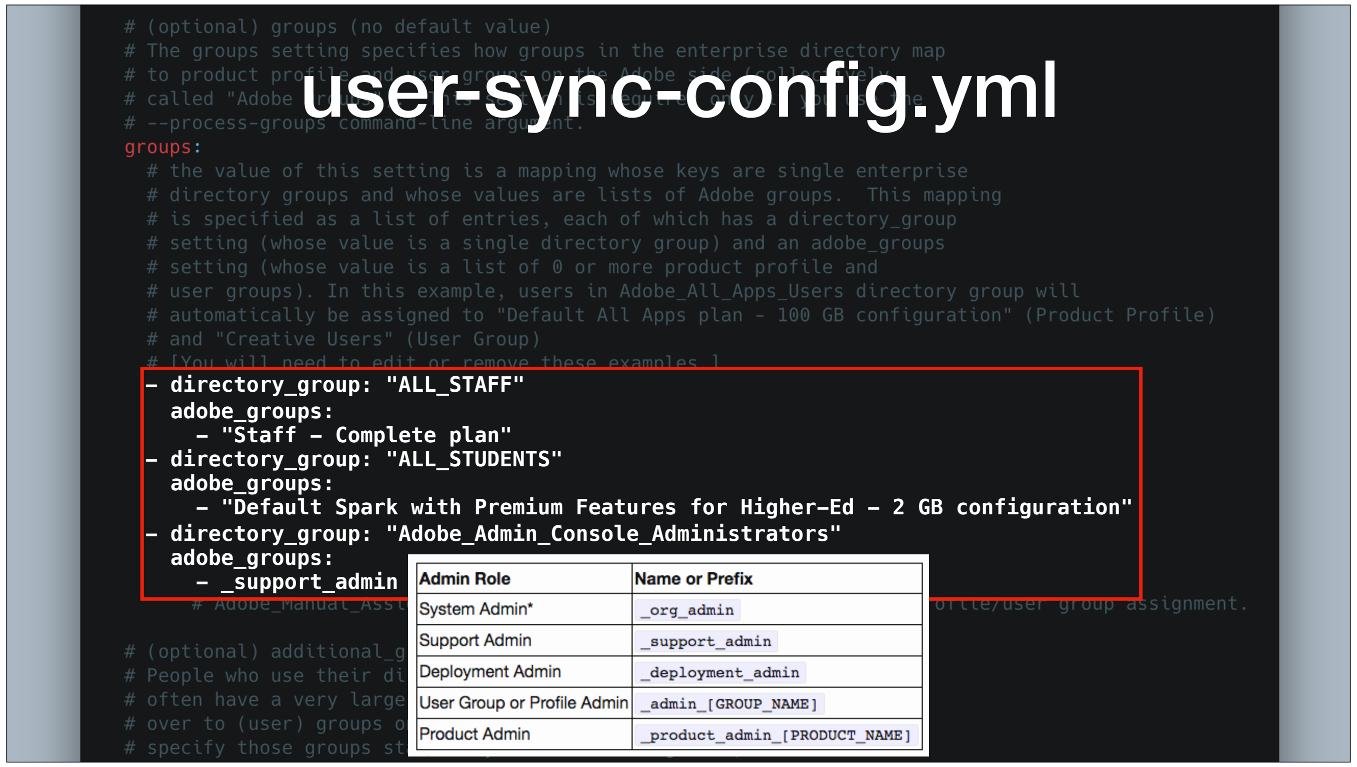


user-sync-config.yml

```
# (optional) groups (no default value)
# The groups setting specifies how groups in the enterprise directory map
# to product profile and user groups on the Adobe side (collectively
# called "Adobe groups"). This section is required only if you use the
# --process-groups command-line argument.
groups:
  # the value of this setting is a mapping whose keys are single enterprise
  # directory groups and whose values are lists of Adobe groups.  This mapping
  # is specified as a list of entries, each of which has a directory_group
  # setting (whose value is a single directory group) and an adobe_groups
  # setting (whose value is a list of 0 or more product profile and
  # user groups). In this example, users in Adobe_All_Apps_Users directory group will
  # automatically be assigned to "Default All Apps plan - 100 GB configuration" (Product Profile)
  # and "Creative Users" (User Group)
  # [You will need to edit or remove these examples.]
  - directory_group: "Acrobat_DC_Pro_Users"
    adobe_groups:
      - "Acrobat DC Pro Users"
  - directory_group: "Adobe_All_Apps_Users"
    adobe_groups:
      - "Default All Apps plan - 100 GB configuration"
      - "Creative Users"
  - directory_group: "Adobe_Manual_Assign_Users"
    adobe_groups:
      # Adobe_Manual_Assign_Users group does not have default product profile/user group assignment.

# (optional) additional_groups (no default value)
# People who use their directory groups for ACLs on the Adobe side
# often have a very large number of groups that they want mapped
# over to (user) groups on the Adobe side.  To avoid having to
# specify those groups statically in their config file, and to
```

Now, this is the bit where you need to tell the User Sync Tool which directory groups get which product profiles (adobe_group means product profile here). You create product profiles in your Admin Console from the products you own. A profile is essentially a group of settings for a product such as which cloud services you're enabling.

user-sync-config.yml



```
# (optional) groups (no default value)
# The groups setting specifies how groups in the enterprise directory map
# to product profile and user groups on the Adobe side (collectively
# called "Adobe groups"). This section is required only if you use the
# --process-groups command-line argument.
groups:
  # the value of this setting is a mapping whose keys are single enterprise
  # directory groups and whose values are lists of Adobe groups.  This mapping
  # is specified as a list of entries, each of which has a directory_group
  # setting (whose value is a single directory group) and an adobe_groups
  # setting (whose value is a list of 0 or more product profile and
  # user groups). In this example, users in Adobe_All_Apps_Users directory group will
  # automatically be assigned to "Default All Apps plan - 100 GB configuration" (Product Profile)
  # and "Creative Users" (User Group)
  # [You will need to edit or remove these examples.]
  - directory_group: "ALL_STAFF"
    adobe_groups:
      - "Staff - Complete plan"
  - directory_group: "ALL_STUDENTS"
    adobe_groups:
      - "Default Spark with Premium Features for Higher-Ed - 2 GB configuration"
  - directory_group: "Adobe_Admin_Console_Administrators"
    adobe_groups:
      - _support_admin
      # Adobe_Manual_Assi                                    ofile/user group assignment.

# (optional) additional_g
# People who use their di
# often have a very large
# over to (user) groups o
# specify those groups st
```

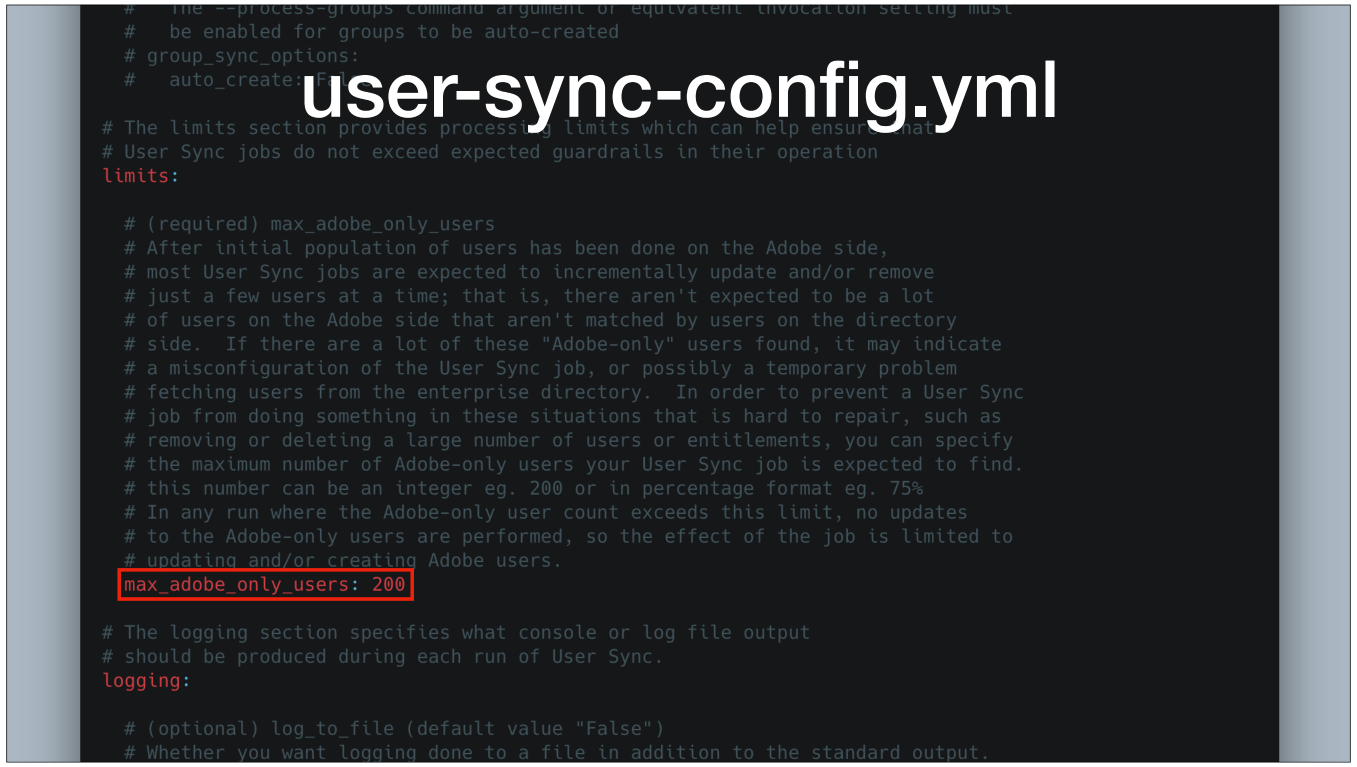
Admin Role	Name or Prefix
System Admin*	_org_admin
Support Admin	_support_admin
Deployment Admin	_deployment_admin
User Group or Profile Admin	_admin_[GROUP_NAME]
Product Admin	_product_admin_[PRODUCT_NAME]

So for us - our all staff AD group is assigned a product profile called Staff - Complete Plan - this contains the all apps and 100GB cloud storage product.

Our ALL_STUDENTS AD group gets the Spark product that comes with 2GB of cloud storage.

Our Adobe_Admin_Console_Administrators group doesn't get a product profile, but it's mapped to a special group called _support_admin. These users will get that Support Administrator role in the Admin Console. You can map all the admin roles except the top level System Admin one.

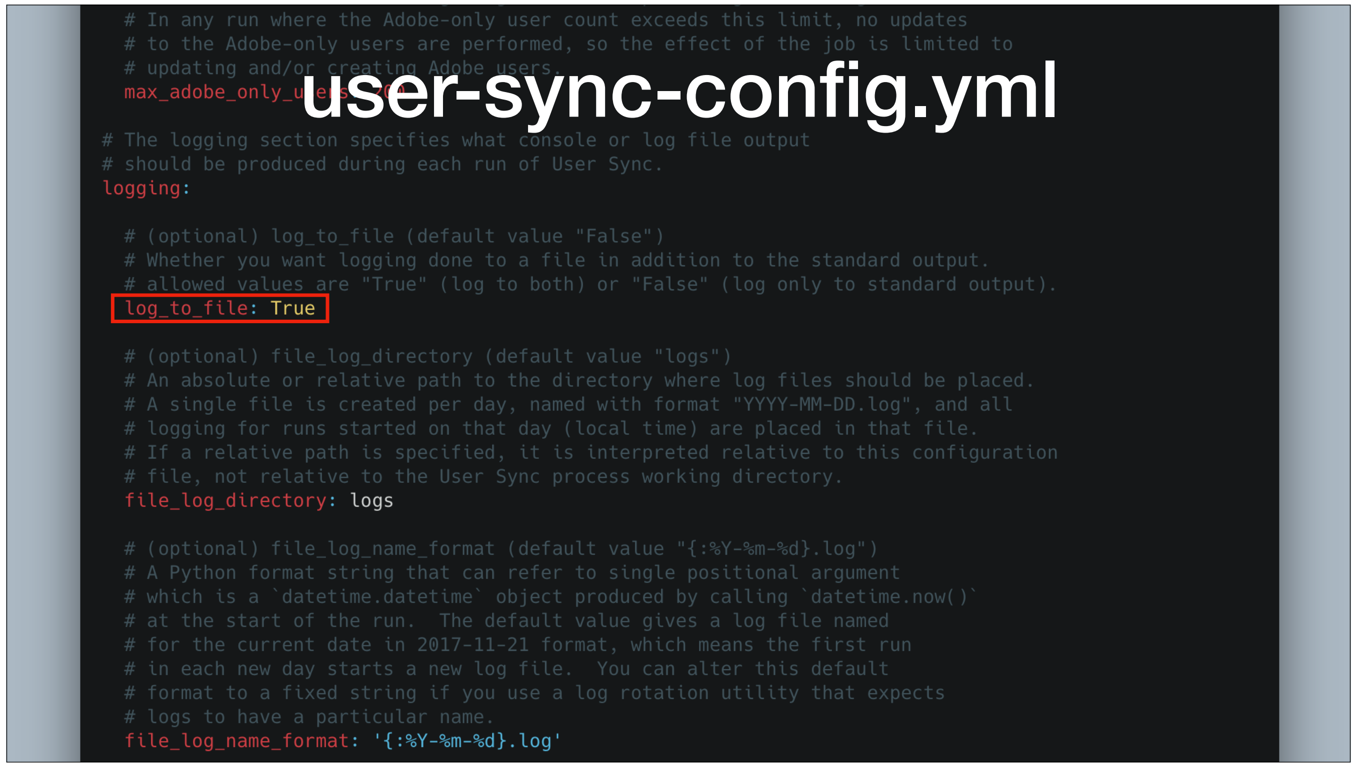
user-sync-config.yml



```
  #   The --process-groups command argument or equivalent invocation setting must
  #   be enabled for groups to be auto-created
  # group_sync_options:
  #   auto_create: Fal

# The limits section provides processing limits which can help ensure that
# User Sync jobs do not exceed expected guardrails in their operation
limits:

  # (required) max_adobe_only_users
  # After initial population of users has been done on the Adobe side,
  # most User Sync jobs are expected to incrementally update and/or remove
  # just a few users at a time; that is, there aren't expected to be a lot
  # of users on the Adobe side that aren't matched by users on the directory
  # side.  If there are a lot of these "Adobe-only" users found, it may indicate
  # a misconfiguration of the User Sync job, or possibly a temporary problem
  # fetching users from the enterprise directory.  In order to prevent a User Sync
  # job from doing something in these situations that is hard to repair, such as
  # removing or deleting a large number of users or entitlements, you can specify
  # the maximum number of Adobe-only users your User Sync job is expected to find.
  # this number can be an integer eg. 200 or in percentage format eg. 75%
  # In any run where the Adobe-only user count exceeds this limit, no updates
  # to the Adobe-only users are performed, so the effect of the job is limited to
  # updating and/or creating Adobe users.
  max_adobe_only_users: 200

# The logging section specifies what console or log file output
# should be produced during each run of User Sync.
logging:

  # (optional) log_to_file (default value "False")
  # Whether you want logging done to a file in addition to the standard output.
```

Pay attention to this one - it’s a safeguard to make sure you won’t blow away a large number of users in the Admin Console in case there’s a misconfiguration in the sync tool or issue reading from your directory server. It won’t touch anything on the Adobe side if the number of users it thinks have changed is greater than this value. You might need to increase it if you expect changes to more users than this during each sync run.

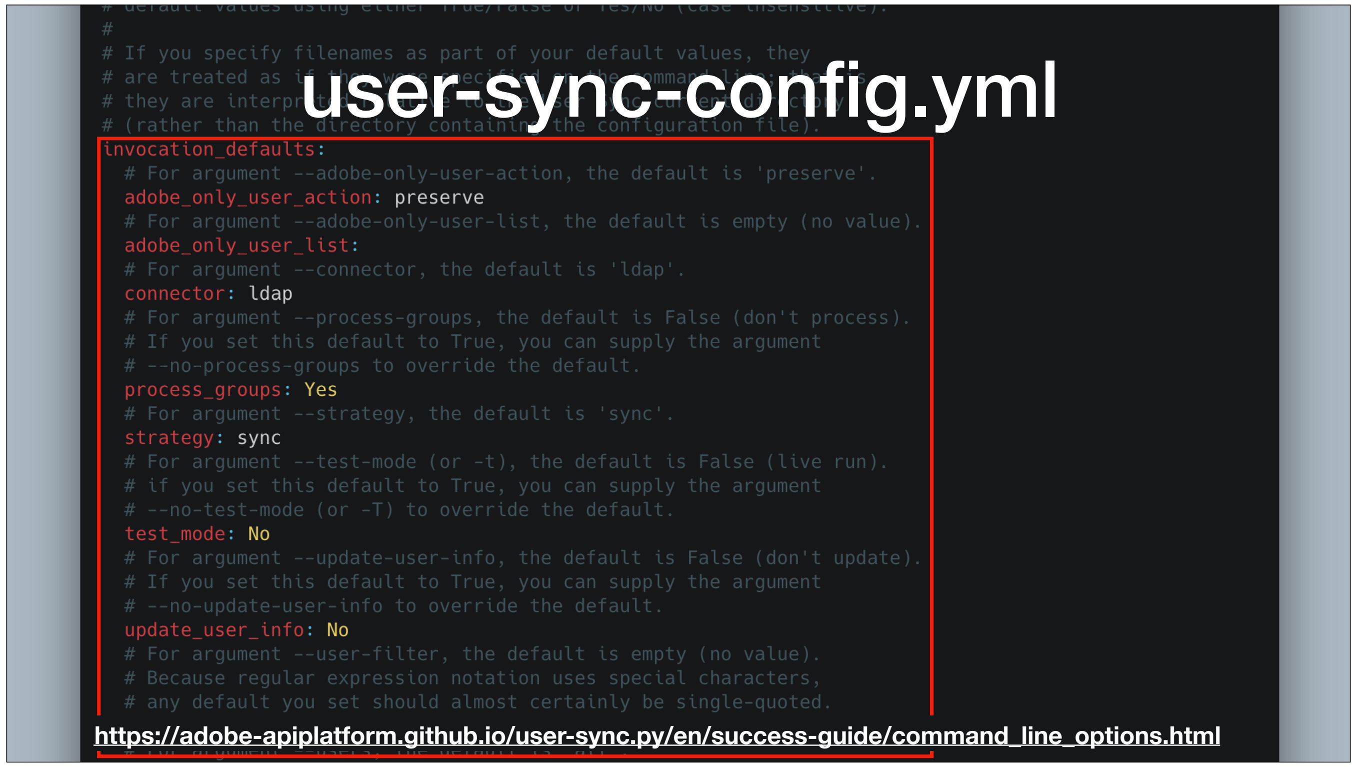
user-sync-config.yml



```
  # In any run where the Adobe-only user count exceeds this limit, no updates
  # to the Adobe-only users are performed, so the effect of the job is limited to
  # updating and/or creating Adobe users.
  max_adobe_only_users: 200

# The logging section specifies what console or log file output
# should be produced during each run of User Sync.
logging:

  # (optional) log_to_file (default value "False")
  # Whether you want logging done to a file in addition to the standard output.
  # allowed values are "True" (log to both) or "False" (log only to standard output).
  log_to_file: True

  # (optional) file_log_directory (default value "logs")
  # An absolute or relative path to the directory where log files should be placed.
  # A single file is created per day, named with format "YYYY-MM-DD.log", and all
  # logging for runs started on that day (local time) are placed in that file.
  # If a relative path is specified, it is interpreted relative to this configuration
  # file, not relative to the User Sync process working directory.
  file_log_directory: logs

  # (optional) file_log_name_format (default value "{:%Y-%m-%d}.log")
  # A Python format string that can refer to single positional argument
  # which is a `datetime.datetime` object produced by calling `datetime.now()`
  # at the start of the run.  The default value gives a log file named
  # for the current date in 2017-11-21 format, which means the first run
  # in each new day starts a new log file.  You can alter this default
  # format to a fixed string if you use a log rotation utility that expects
  # logs to have a particular name.
  file_log_name_format: '{:%Y-%m-%d}.log'
```

By default, output from the UST is logged to a file and you can set things here like the directory, file name format and so on. Beware that the UST doesn't rotate these. For me, I disable this and instead redirect output from the command itself to a text file so it only logs the most recent run. Then we use a monitoring tool called SCOM to monitor that file and generate an alert for words like WARNING and ERROR

user-sync-config.yml

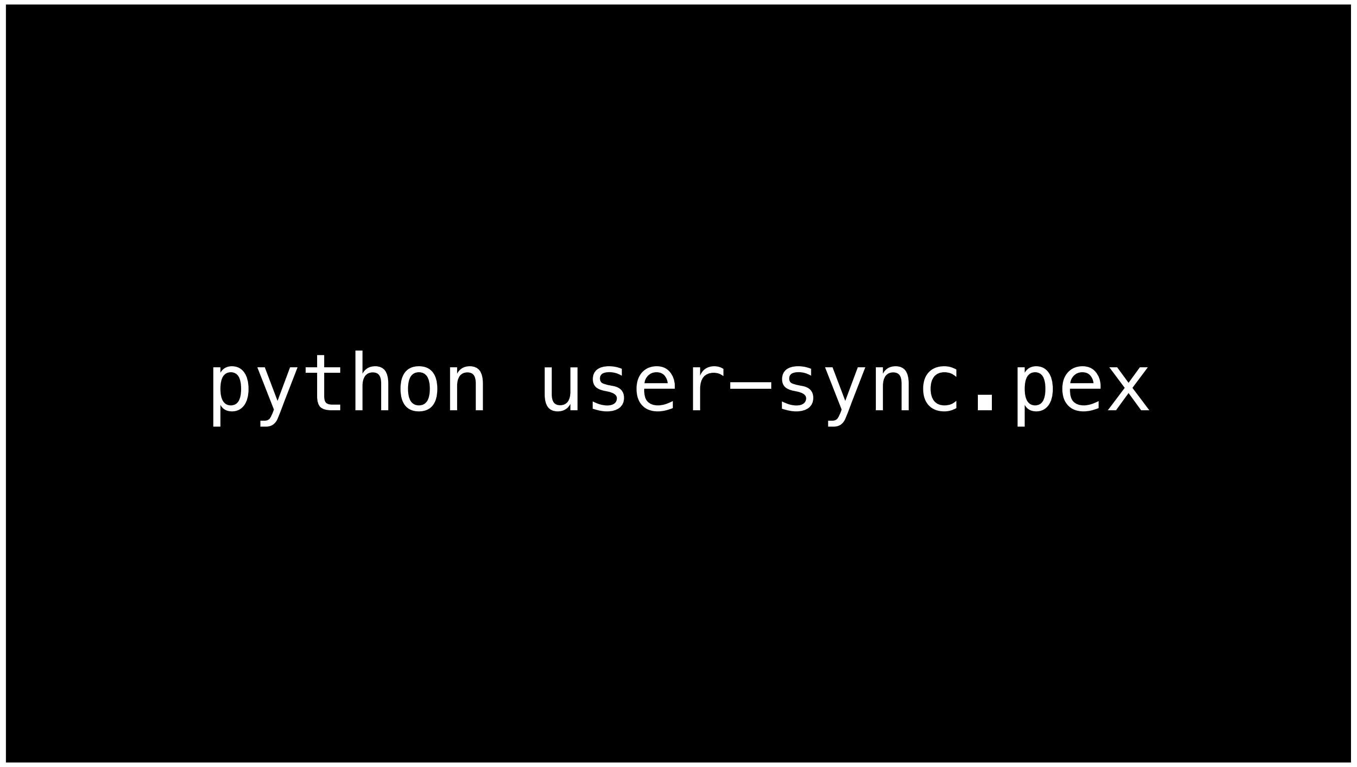


```
# default values using either True/False or Yes/No (case insensitive).
#
# If you specify filenames as part of your default values, they
# are treated as if they were specified on the command line; that is,
# they are interpreted relative to the User Sync current directory
# (rather than the directory containing the configuration file).
invocation_defaults:
  # For argument --adobe-only-user-action, the default is 'preserve'.
  adobe_only_user_action: preserve
  # For argument --adobe-only-user-list, the default is empty (no value).
  adobe_only_user_list:
  # For argument --connector, the default is 'ldap'.
  connector: ldap
  # For argument --process-groups, the default is False (don't process).
  # If you set this default to True, you can supply the argument
  # --no-process-groups to override the default.
  process_groups: Yes
  # For argument --strategy, the default is 'sync'.
  strategy: sync
  # For argument --test-mode (or -t), the default is False (live run).
  # if you set this default to True, you can supply the argument
  # --no-test-mode (or -T) to override the default.
  test_mode: No
  # For argument --update-user-info, the default is False (don't update).
  # If you set this default to True, you can supply the argument
  # --no-update-user-info to override the default.
  update_user_info: No
  # For argument --user-filter, the default is empty (no value).
  # Because regular expression notation uses special characters,
  # any default you set should almost certainly be single-quoted.
```

https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/command_line_options.html

When you run the UST, you'd usually specify some parameters after the command. If you don't, it'll use the ones set under invocation_defaults. That page tells you all about the parameters and what they do.

So… you've set everything up and it's time to run the tool!

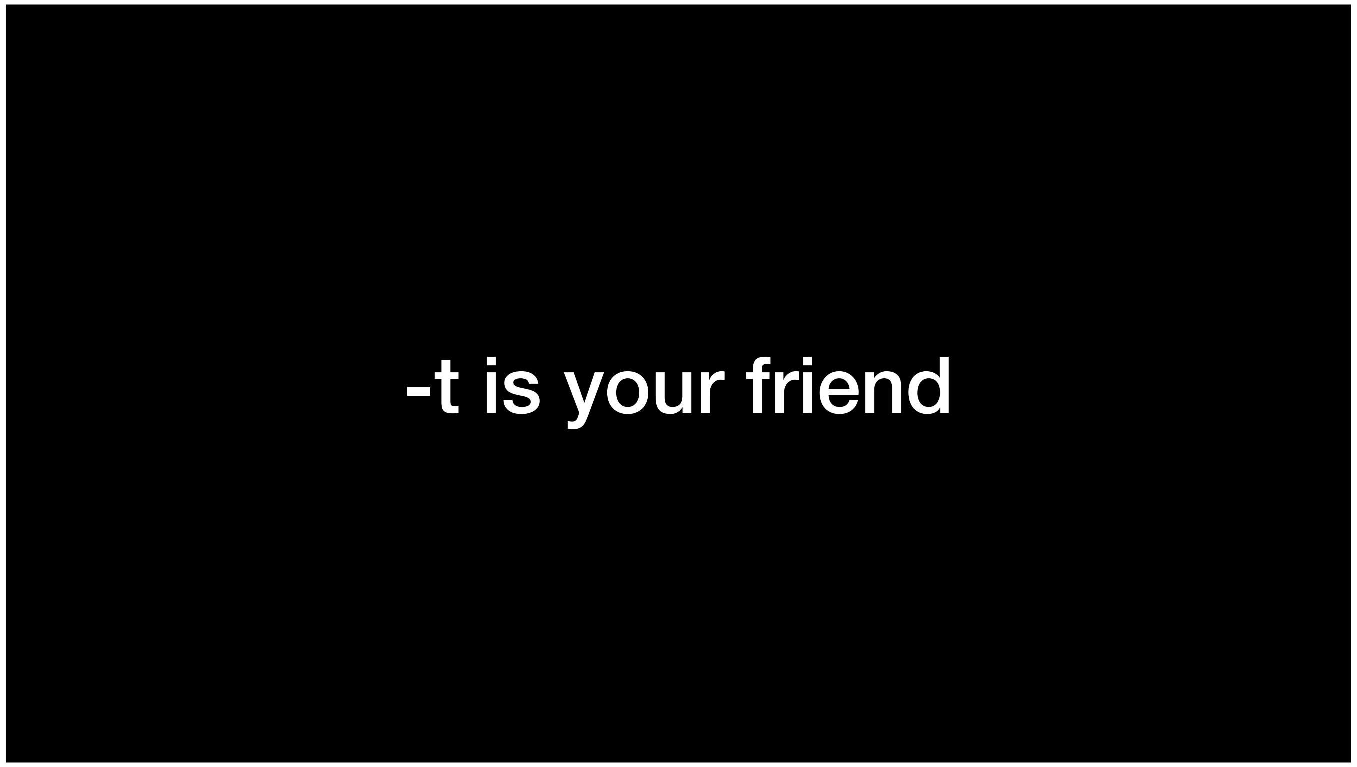


```
python user-sync.pex
```

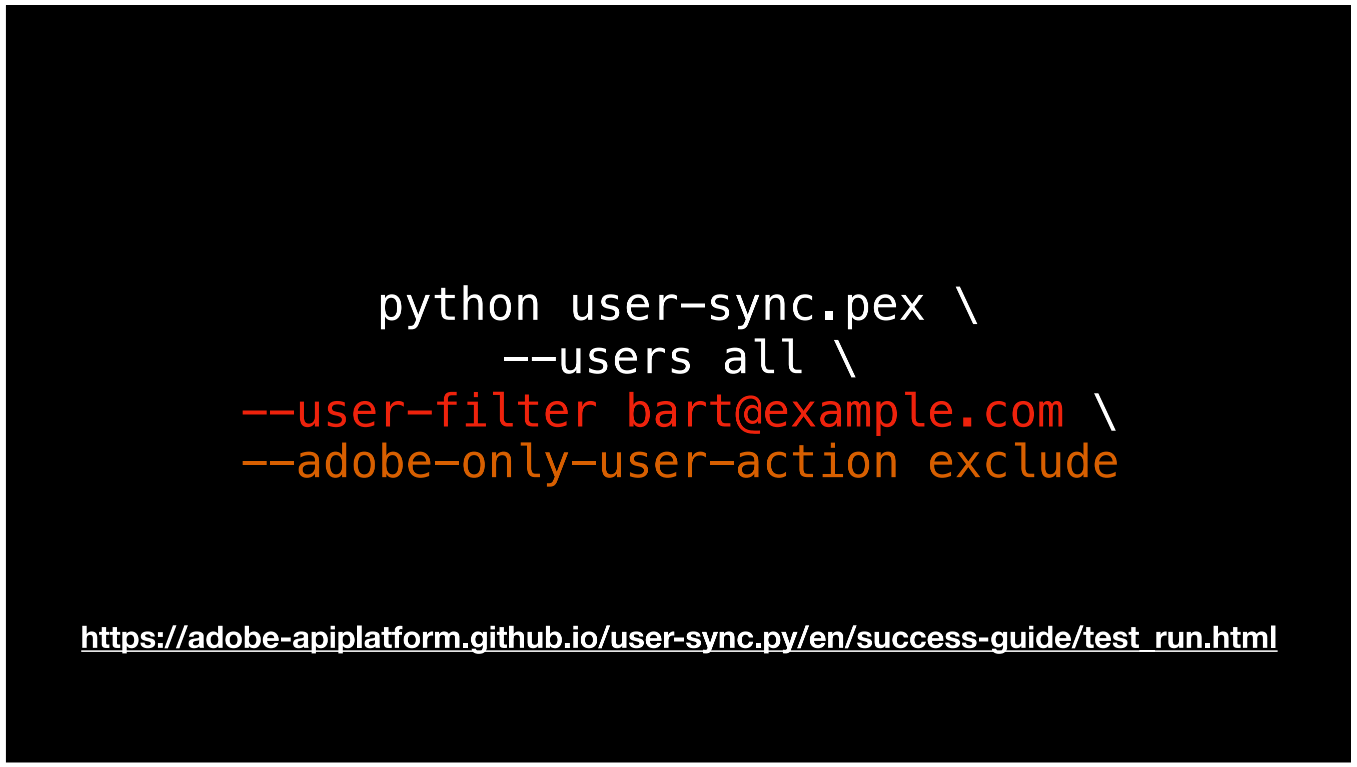
So.. it’s time to run the tool…

Testing

You really should have tested it….


-t is your friend

Use the -t or --test-mode parameter, look at the logs.



```
python user-sync.pex \
--users all \
--user-filter bart@example.com \
--adobe-only-user-action exclude
```

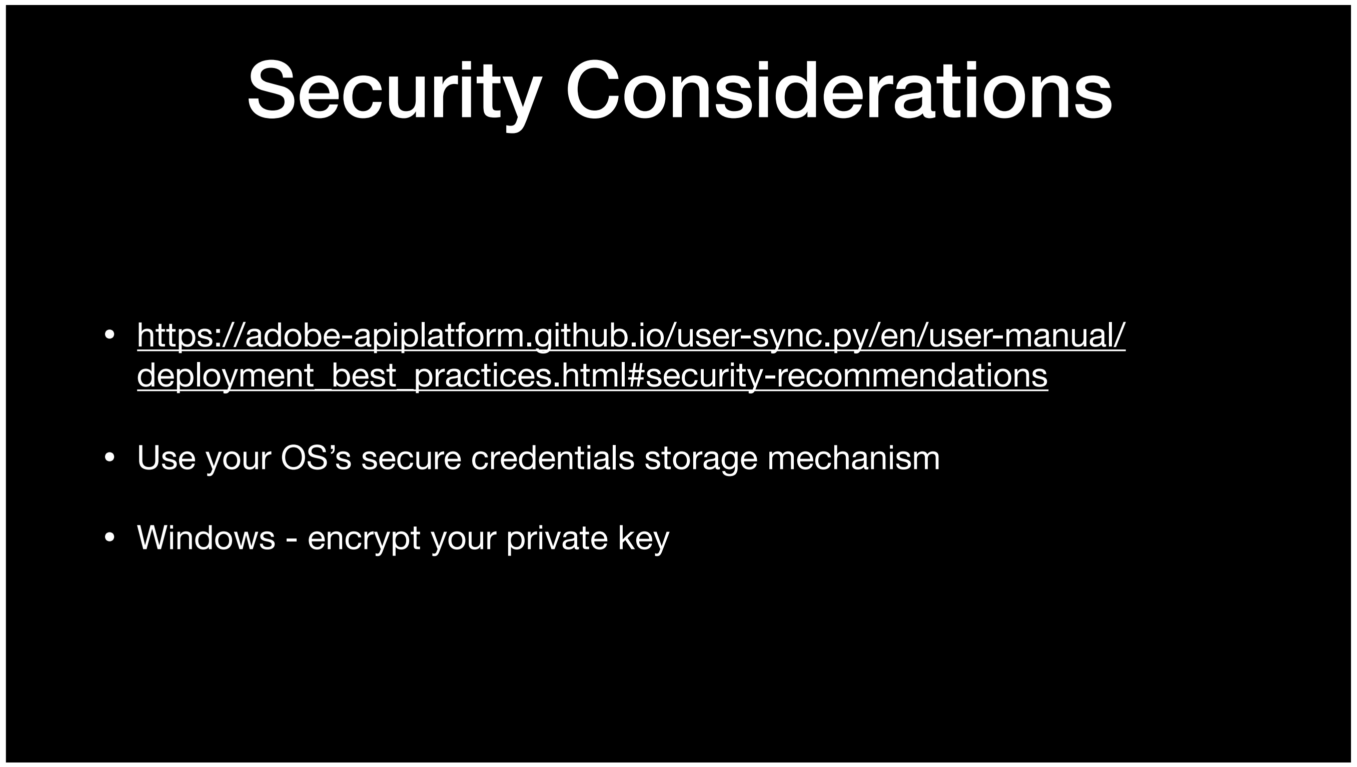
https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/test_run.html

When you do need to try out a live change for the first time, filter it against one user and exclude all your Adobe side users from any actions. Follow the advice on this page!

Profit!

Profit!

Here are some more things to think about.



Security Considerations

- https://adobe-apiplatform.github.io/user-sync.py/en/user-manual/deployment_best_practices.html#security-recommendations
- Use your OS's secure credentials storage mechanism
- Windows - encrypt your private key

You saw that all the credentials for the directory server and UMAPI are stored in plain text in those config files, which is bad!

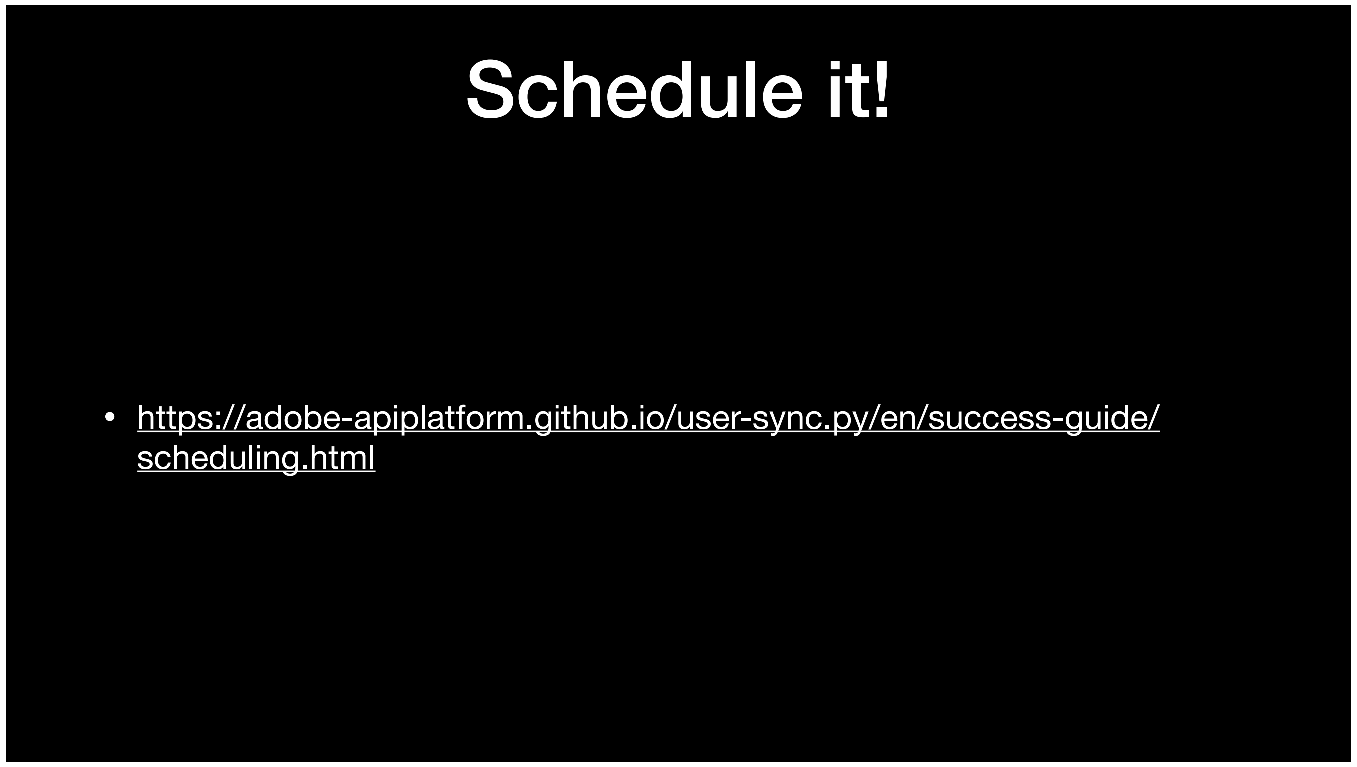
Read that page and make use of your OS's secure credentials storage then make sure you edit those config files so that they point to that in order to retrieve them. This is also detailed in the comments of those example config files.

For windows, also encrypt your private key and put the passphrase in the Credentials Manager.

Account Deletion

- Consider your deletion strategy for users who leave
- Deleted users lose their cloud storage as well
- https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/decide_deletion_policy.html
- https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/command_line_options.html

Think about account deletion
Think about GDPR and holding onto PII in the admin console as a consequence of preserving.
This also depends on how you handle accounts in your org when folks leave - e.g. if disable them, move them into a “graveyard” directory group?
This page has some guidance on it.

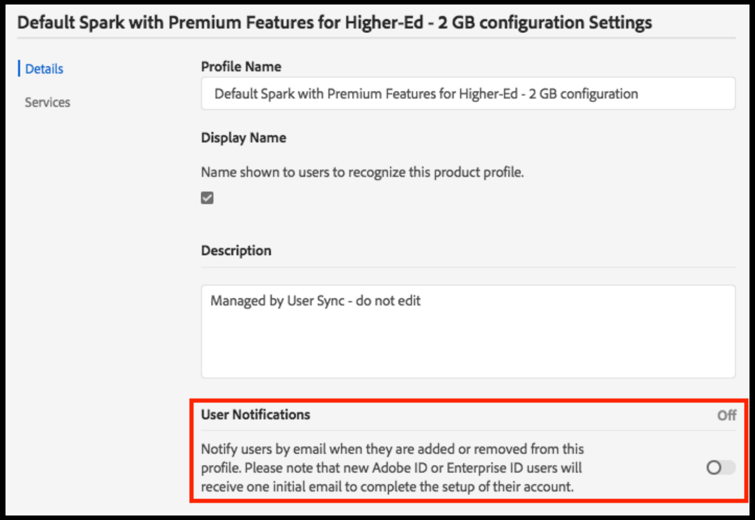


Schedule it!

- https://adobe-apiplatform.github.io/user-sync.py/en/success-guide/scheduling.html

You’ll want to have this run on some kind of automated schedule - check out this page for plenty of examples on what to do.
We run the sync tool on two servers at two different sites for a bit of redundancy - one syncs at 1am and the other syncs at 1pm.

Emails?


Default Spark with Premium Features for Higher-Ed - 2 GB configuration Settings

Details

Services

Profile Name

Default Spark with Premium Features for Higher-Ed - 2 GB configuration

Display Name

Name shown to users to recognize this product profile.

☑

Description

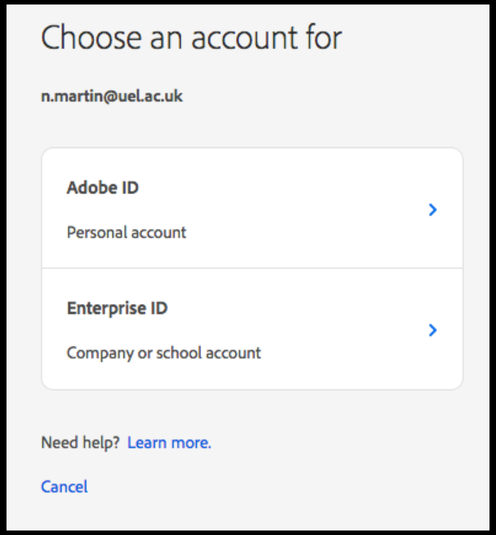
Managed by User Sync - do not edit

User Notifications — Off

Notify users by email when they are added or removed from this profile. Please note that new Adobe ID or Enterprise ID users will receive one initial email to complete the setup of their account.

The admin console will send users emails when their product profiles change. You can turn this off for each product profile in its settings.

ID conflicts


Choose an account for

n.martin@uel.ac.uk

Adobe ID

Personal account

Enterprise ID

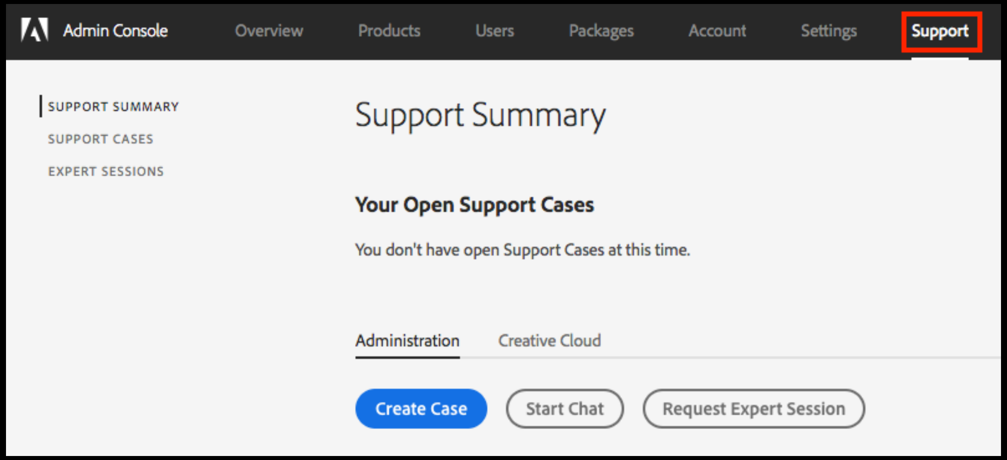
Company or school account

Need help? Learn more.

Cancel

If your users have their own Adobe IDs but they're tied to their organisation's email accounts, they'll see this when they try to sign in. So you need to make sure they're aware and know what to do.

Get Support!



Support Summary

Your Open Support Cases

You don't have open Support Cases at this time.

Finally, because of the changes around Shared Device Licensing, Adobe know more people will be moving to this so they're making an effort to help us out. They might have even reached out to you already.
If they haven't, you should contact your account manager/rep, or open a case/request an expert session through the Support tab in your Admin Console.

Thank you!

And that’s it!