

|  |  |
| --- | --- |
|  |  |



**Deployment Guide**

**May 2020**

**Contents**

[1 Setting up Web AppBuilder and Waahi for the first time 3](#_Toc39051580)

[1.1 Download Web AppBuilder for ArcGIS (Developer Edition) SDK 3](#_Toc39051581)

[1.2 Download and unzip the Waahi application 3](#_Toc39051582)

[1.3 Run the deployment app 3](#_Toc39051583)

[1.4 Troubleshooting 4](#_Toc39051584)

[1.5 Additional setup 4](#_Toc39051585)

[2 Upgrade Web AppBuilder to a newer version 5](#_Toc39051586)

[2.1 Download the new version of WAB 5](#_Toc39051587)

[2.2 Deploy Waahi to the new version of WAB 5](#_Toc39051588)

[2.3 Migrate existing apps to the new version of WAB 5](#_Toc39051589)

[3 Upgrade Waahi to a newer release 6](#_Toc39051590)

[4 Service Deployment (Optional) 6](#_Toc39051591)

[4.1 Access to services 6](#_Toc39051592)

[4.2 Print Service 6](#_Toc39051593)

[4.3 Share Service 12](#_Toc39051594)

# Setting up Web AppBuilder and Waahi for the first time

* 1. Download Web AppBuilder for ArcGIS (Developer Edition) SDK

Download the version of Web AppBuilder (WAB) required and that is compatible with the latest Waahi release which aren’t normally far behind the latest version of WAB. The version for which a specific Waahi release has been tested against is in the first part of the release name, for example 2.12.x.x means it was developed and tested against version 2.12 of WAB. Find different versions of WAB ready for download at this page:

<https://developers.arcgis.com/downloads/apis-and-sdks?product=web-appbuilder>

Follow this guide to get WAB running on your machine if it hasn’t been set up already:

<https://developers.arcgis.com/web-appbuilder/guide/getstarted.htm>

* 1. Download and unzip the Waahi application

Once the WAB instance is ready, download and unzip the Waahi-WAB deployment app to a separate folder. Once the source code for the Waahi instance has been downloaded, the Waahi deployment app can be used to copy relevant files into your Web AppBuilder (WAB) instance for a typical Waahi setup.

For example, you will have the WAB folder with its contents, and the Waahi app with its contents in another folder:

WebAppBuilder-dev

|---- client

|---- docs

|---- server

Waahi-WAB

|---- deploy\_app

|---- stemapp

|---- node\_x64.exe

|---- node\_x86.exe

|---- README.md

|---- ui.bat

* 1. Run the deployment app

As an extra precaution, make sure that WAB is not running while you follow the next steps. Run "ui.bat". A browser should open and display the deployment console (on <http://localhost:3347/>). Run it in Google Chrome, not IE. If you encounter issues check the port setting in the ui.bat file.

Once running:

* Use the "Existing Web AppBuilder location" input and click **set** to update the location of your relevant or new WAB install.
* Select the deployment configuration, typically this will be "Waahi".
* Use the "Deploy" button to copy relevant files.

A file watcher is also available, which monitors files in the "stemapp" source folder for changes. This is typically used during development only.

* 1. Troubleshooting

If you receive errors, check that Web AppBuilder runs by itself, before any Waahi code has been added. This will help pinpoint the problem.

Also check file permissions - Node.js needs to be able to modify files in the Web AppBuilder and Waahi-WAB directories.

* 1. Additional setup

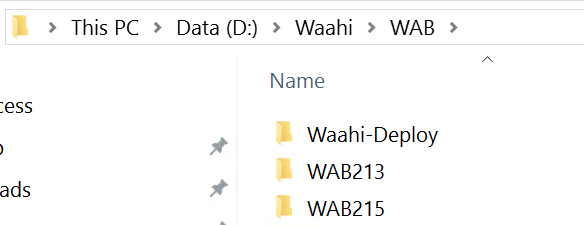
Additional Waahi components that can be optionally installed are:

* Waahi-Services (web services) includes a .NET Core web service for the Share Plus widget, as well as Python geoprocessing services for Print and Data Export widgets.  
  [Waahi-Services repo](https://github.com/GeographicBusinessSolutions/Waahi-Services)
* Waahi-Gallery is a minimalistic landing page for your ArcGIS Enterprise or ArcGIS Online users, allowing Waahi and other deployed applications to be easily discovered.  
  [Waahi-Gallery repo](https://github.com/GeographicBusinessSolutions/Waahi-Gallery)

# Upgrade Web AppBuilder to a newer version

* 1. Download the new version of WAB

Download and unzip the new version of WAB in parallel to the previous version that was set up (don’t replace the previous version).



If the new WAB’s startup.bat file is run, it will ask for the portal URL and an App ID. Use the same portal and app ID that was used for the previous WAB instance. The new WAB version will still run on the same port and URL. One of these WAB instances can now run at a time by running its Startup.bat file and going to the WAB URL in the browser.

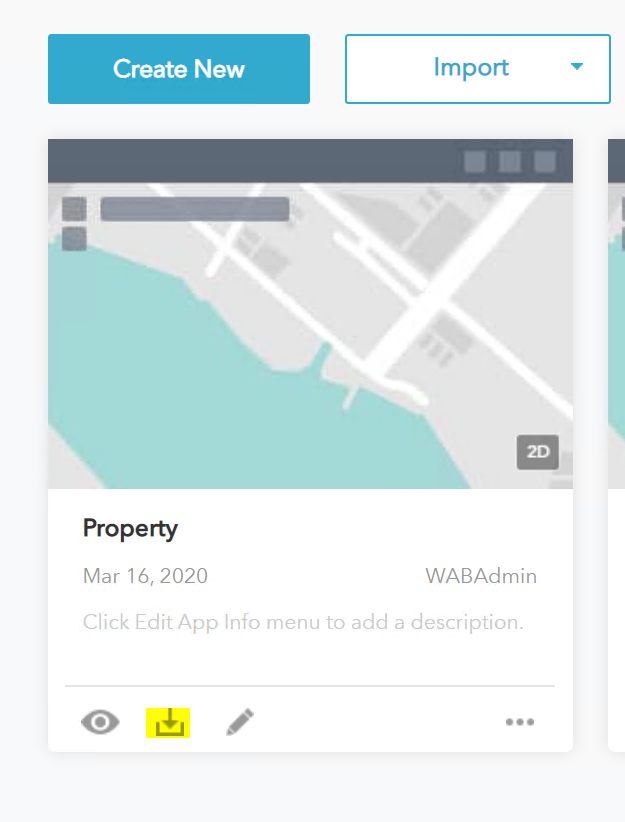
* 1. Deploy Waahi to the new version of WAB

Run the Waahi deployment app again and set the location to the new WAB instance. Waahi is now deployed to this WAB.

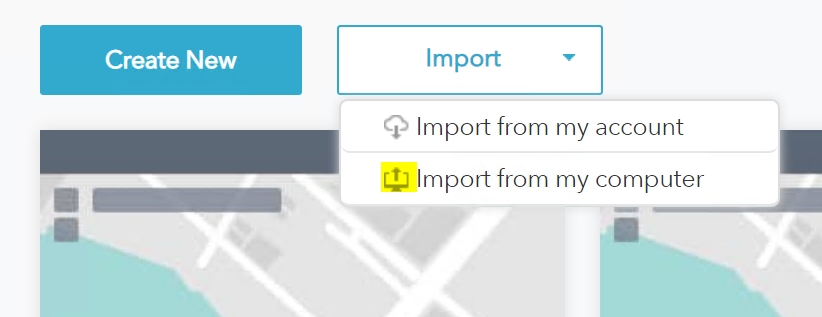
* 1. Migrate existing apps to the new version of WAB

Migrating existing apps from an older version of WAB to the newer version entails the following:

* + 1. Open the previous WAB instance and download the relevant apps (note that the old and new WAB cannot be running simultaneously, because they are on the same port, 3344, by default):



* + 1. Close that instance of WAB and open the latest one that you want to migrate the apps to. Import from Computer, and point to the downloaded apps (they should still be zipped). Import the apps one at a time.



Notes

The Waahi release name should tell you the latest version of WAB against which Waahi was fully tested, for example 2.12.x.x means it was developed and tested against version 2.12 of WAB. Make sure that the version of Waahi that is being used has been tested against your version of WAB.

Some widget reconfiguration might be needed, depending on the changes in the new WAB version. We always try to highlight any reconfiguration needed in the release notes.

# Upgrade Waahi to a newer release

Follow the steps in sections 1.2 and 1.3.

The Deployment app will deploy the latest code into Web AppBuilder, replacing outdated files from the previous release that was implemented. All existing applications created in WAB will also be updated with the latest code, therefore, when configuring applications, the latest widgets and functionality will be applied. In some cases, a widget might have to be reconfigured if big changes to that widget has taken place.

# Service Deployment (Optional)

* 1. Access to services

Python geoprocessing services are used alongside Web AppBuilder widgets such as Print and Share and should be published to an ArcGIS Server instance.

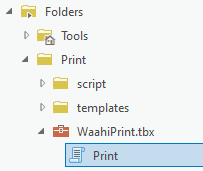
Source code for Waahi services are available here, please contact GBS for access:

<https://github.com/GeographicBusinessSolutions/Waahi-Services/releases>

* 1. Print Service

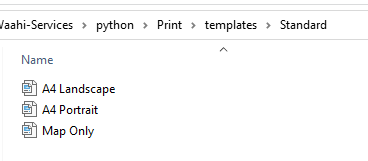
The printing service is used by the Waahi Print (Print Plus) widget for web map printing and has additional template capability.

The print tools for ArcGIS Pro are available in the services repository. Extract the services folder and browser to /python/Print. Add this folder to ArcGIS Pro.

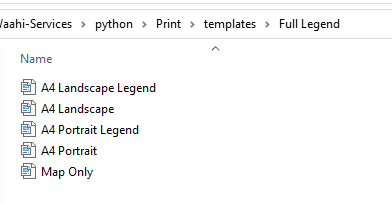


In this folder, there are default layouts that can be used for web map printing. You can use your own templates instead, by creating layouts in ArcGIS Pro. The layouts should be PAGX files.

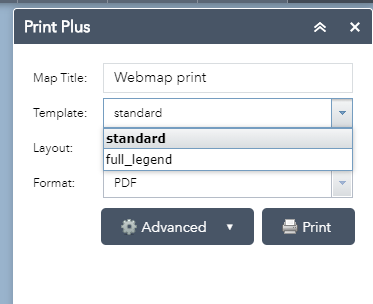
You can have multiple templates, and a template can contain multiple layouts. In the following template, “standard”, there are layouts for A4 Landscape, A4 Portrait, and Map Only.

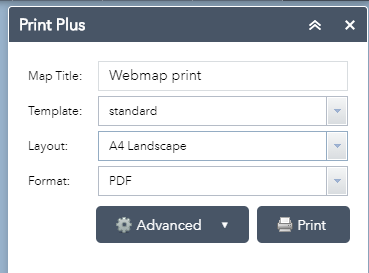
****

Full page legends can be used by adding “legend” to the end of the layout name. These will be added as an additional page when printing. The “Legend” layouts are optional, a legend could be included inside the “A4 Portrait” and “A4 Landscape” layout files instead.

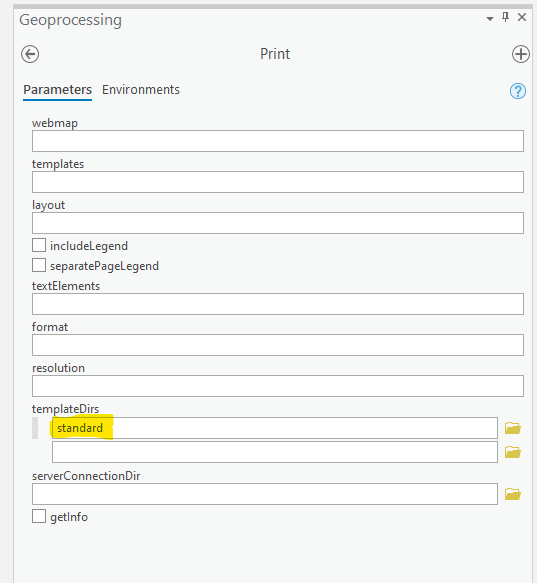


To demonstrate, when published, the templates and associated layouts can be selected by a print user:

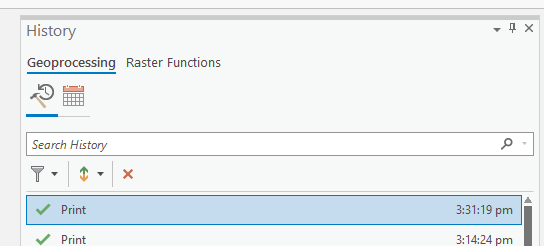


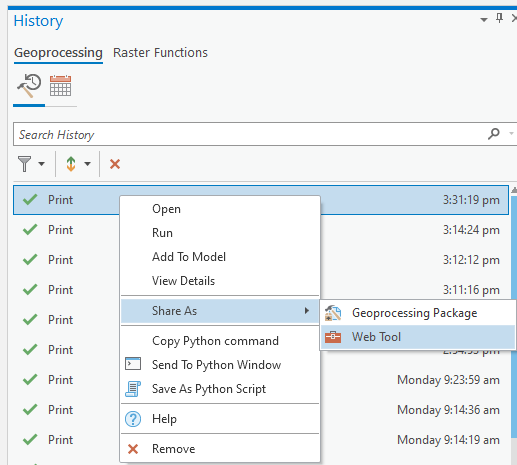


When your layouts have been created and are ready to go, run the Print tool. Your template folder, containing PAGX files, should be selected as the “templateDirs” parameter.

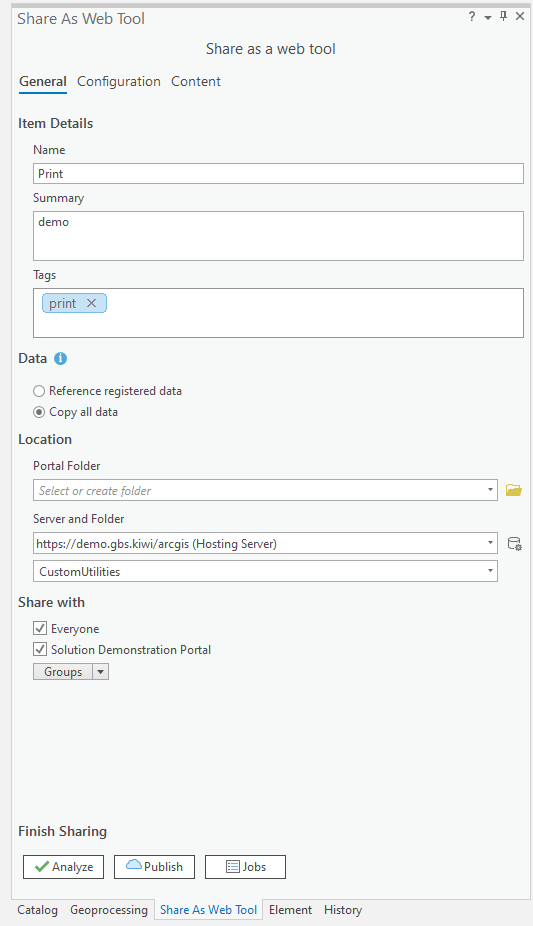


View the completed tool in your geoprocessing history, and select “Share as Web Tool”

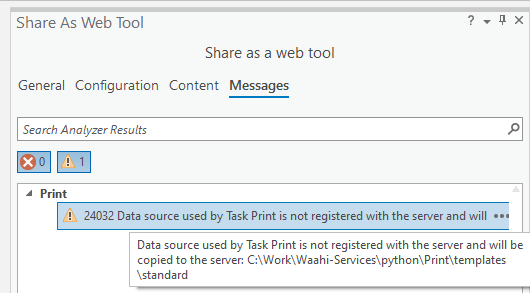




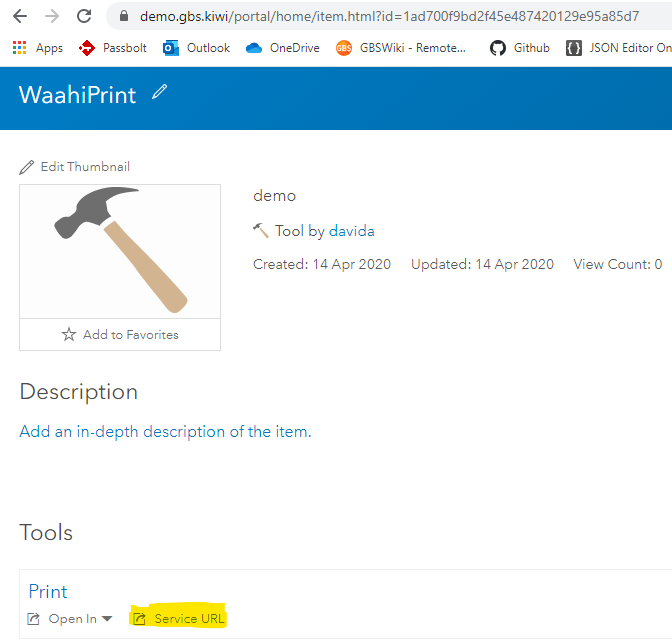
Fill in the relevant parameters. In the configuration tab, make sure execution mode is set to “Asynchronous”. Publish the tool.



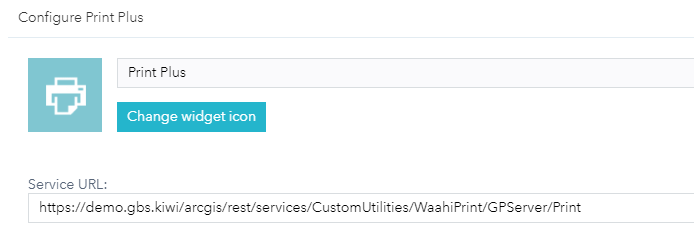
The following message is expected, as the print templates will be uploaded during publishing.



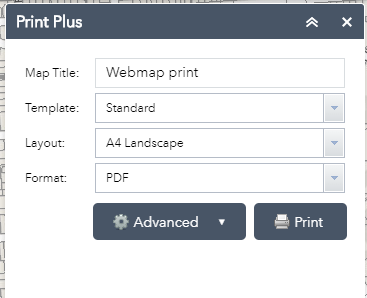
When published successfully, the URL can be configured in the Print widget in Web AppBuilder. Find the URL from the Portal item:



In Web AppBuilder, while configuring the Print Plus widget, add the Service URL:



If the widget has the correct service URL and was configured correctly, the user will see this when opening the widget from the app:



* 1. Share Service

The RESTful share service is used by the Waahi Share (Share Plus) widget and allows saving map settings (extent and graphics) to a database, for retrieval by URL (using a unique code). It uses an embedded SQLite (file-based) database by default but can be switched to use SQL Server. SQL Server is recommended for high-use applications.

* + 1. Extract the compiled publish output files

This service is compiled in Visual Studio, but for deployment, the /PublishOutput distribution folder can be used. Download the repo, and browse to:

/dotnet/share/PublishOutput

Extract the contents of PublishOutput.zip and move it to a location ready for setup in IIS (such as C:\inetpub\wwwroot\s).

#### Install .NET Core IIS

Make sure IIS can run .NET Core applications, this includes installing the .NET Core Windows Server Hosting bundle:

<https://docs.microsoft.com/en-us/aspnet/core/host-and-deploy/iis/?tabs=aspnetcore2x&view=aspnetcore-2.1#install-the-net-core-hosting-bundle>

Note: the current .NET Core target version is v3.1.

#### Create an IIS site

Create an IIS site pointing to the deployment folder. This should have an application pool with a .NET CLR version of "No Managed Code". We recommend using a short name, so that shared URLs are as short is possible; for example: "<https://secure.gbs.co.nz/s>".

#### Create the database: SQL Server (optional)

The compiled service uses a SQLite database by default (share\_data.db), which should work as-is. We recommend switching to SQL Server for larger sites, or where there will be frequent or high use of the share service. To switch to SQL Server:

* Set up a database
* Create a table "gisco\_share\_service", using the SQL in: <https://github.com/GeographicBusinessSolutions/Waahi-Services/blob/master/dotnet/Share/SQL/create_share_data_table.sql>
* You will then need to update the DefaultConnection setting in appsettings.json.

#### Move the database: SQLite (optional)

The compiled service uses a SQLite database in the PublishOutput folder by default (share\_data.db). We recommend moving this to a location outside of the web service, to remove the risk of overwriting it when upgrading. You will then need to update the DefaultConnection setting in appsettings.json.

#### Configure the application

Configure the application settings in appsettings.json. This includes:

DefaultConnection // connection to the SQLite or SQL Server database set above

BaseUrl // the URL to the service, including the host name and site name,

// example: https://maps.mysite.co.nz/s?

#### Updating an existing deployment

To update a service, the contents of the PublishOutput folder can be updated (see step 4.3.1).

IMPORTANT: avoid overwriting share\_data.db! If using the default SQLite database file "share\_data.db" as-is, copying updated service contents will overwrite your database and existing share data will be lost. We recommend moving the .db file outside of the service folder: see step 5 above. Alternatively, backup the file and restore it after copy.

#### Troubleshooting

* The first step to debugging issues is enabling logging: set the stdoutLogEnabled property to "true" in the web.config file. Logs will by default be created in /logs.
* If the application is not running, you may need to explicitly set the path to the .NET core executable in the web.config file. Look for the "processPath" setting. The dotnet executable is commonly located here: "C:\Program Files\dotnet\dotnet.exe"
* The dotnet version must be at 3.1 or higher. Check the version in the command prompt using: dotnet --version
* If using SQLite, the service must have permissions to write to the database (share\_data.db). Check the windows permissions for this file.

#### Widget configuration in WAB

Once the share service is deployed, the Waahi Share Widget (Share Plus) can be configured in Web AppBuilder.

* If using the default Waahi app, remove the default WAB Share widget
* Add the Share Plus widget
* Add the Share service URL into the settings page. For example: <https://secure.gbs.co.nz/s>

#### Cleaning the share data table

Depending on viewer use, a large number of records can be created in the database. Each record is saved with a "last used" date. We recommend cleaning out older records on a regular basis; for example, if last used date is older than a year.

* DB Browser for SQLite can be used for SQLite. Open file "share\_data.db".
* SQL Server Management Studio is recommended for SQL Server.
* Browse to the database table "gisco\_share\_data".
* Delete records where date is older than specified interval (for example, one year).