Web Package Manager and the Web Packages node in the Solution Explorer

With Servoy 8.1 the developer has a Web Package Manager through which Servoy ships a few packages instead of bundling those with the core. Currently they are '12grid', 'Bootstrap Components' and 'Servoy Extra Components'. This way these packages can have there own release cycle independently of Servoy releases (As long as the package doesn't need extra core functionality of the Servoy product)

The WPM can be opened through the Web Packages node on the Solution Explorer, if installed then the packages will be installed in that same folder in the selected solution. So those packages are not stored under the resources anymore but are more tied to the solution. And can be pushed to the git or svn repository together with the other things of the solution.

> ⚠ Active solution and its modules should not have same web package multiple times, should have it only once in either main solution or module. Best practice would be to include it in some module and reference the module from all solutions that use the library.

The above Servoy packages are all hosted on github, as an example see: https://github.com/Servoy/bootstrapcomponents. Those can also be checked out and then used as a Web Package Project (where the solution will have a link to that project instead of a zip file in the Web Packages folder)

3th party developers can also contribute component or service packages through WPM if they want to. Such a package should have a public accessible webpackage.json file, this is currently the content of the bootstrap components:

```
{
    "name":"bootstrapcomponents",
    "displayName":"Bootstrap Components",
    "packageType": "Web-Component",
    "icon": "https://upload.wikimedia.org/wikipedia/commons/thumb/e/ea/Boostrap_logo.svg/1020px-Boostrap_logo.
svg.png",
    "description": "A set of components based on the bootstrap css, intended to be used together with the 12grid
layout package (please install that package as well). You can build a responsive layout for your form using this
combination or components and layouts.",
    "wikiUrl": "https://github.com/Servoy/bootstrapcomponents/wiki",
    "sourceUrl" : "https://github.com/Servoy/bootstrapcomponents",
    "releases" : [{"version":"1.0.1","url":"https://github.com/Servoy/bootstrapcomponents/releases/download/v1.
0.1/bootstrapcomponents.zip"},
                  {"version":"1.0.2","url":"https://github.com/Servoy/bootstrapcomponents/releases/download/v1.
0.2/bootstrapcomponents.zip", "servoy-version":"8.1", "dependency":"12grid#>=1.0.2"},
                  {"version":"1.0.3","url":"https://github.com/Servoy/bootstrapcomponents/releases/download/v1.
0.3/bootstrapcomponents.zip", "servoy-version":"8.1", "dependency":"12grid#>=1.0.3"}]
}
```

- name: the symbolic name of the component should be equal to the Bundle-SymbolicName: bootstrapcomponents in the manifest.mf of the package
- displayName: the name the users sees in the WPM
- icon: the icon that is shown in the WPM
- description: a small piece of text explaining what this package provides and wants.
- wikiUrl/sourceUrl/sampleUrl: external links that the user can click on for various
- releases: this is an array of object that describes one release, one release has:
    - version: the version of this release
    - url: the url that points to a zip file of this release
    - servoy-version: the minimal version of servoy that is required for this version
    - dependency: a comma separated list of names#semantic_version of other packages and there version number that must also be there. (these will be auto installed)

Through github making a release has the following steps:

1> make a new release through the ui of github

2> download the sources.zip

3> unzip this release and remove the webpackage.json and .project file and zip so that the root is the real root (without that extra indirection)

4> attach this file to the release as another binary, and the raw url point to that zip file.

5> alter the webpackage.json file (of the master branch, that is used by the servoy index) and add the new version to it.

6> alter the manifest.mf file and up the version, so that the source is now one version higher then the actual shipping one.

Servoy maintains an index file that has pointers to that webpackage.json, so the only thing a 3th party needs to provide is the is raw file to the master version of that webpackage.json file (that will be constantly updated). For bootstrap this is the link:

https://raw.githubusercontent.com/Servoy/bootstrapcomponents/master/webpackage.json

A web package can either be used as an archive (zip) or as a referenced project. Referenced project is now preffered way to include a web package reference. In Servoy 8.2 , you can also export the used packages in solution export. This is used for developer export/import only, so you do not need to reinstall packages after each import.