Developing packages for NGClient2

[workspace]/.metadata/.plugins/com.servoy.eclipse.ngclient.ui/target

Is the working directoy of the NGClient2 sources,
the package.json will be adjusted by Servoy depending on the web packages you have installed for the active solution.

Then Servoy will run in developer "npm run build_debug_nowatch" script on that to generate source mapped application for easier debugging in the developer.
It will only run that when it sees it needs to run "npm install webpackage" for a webpackage that is not installed.

"Copy the NGClient2 sources" menu entry in the Solex will overwrite the sources in that target dir and trigger a npm install again for all the packages and will run once a npm build
This is then a clean source install (you copy the sources of Servoy itself right over that to start clean). There is also an question to also use the "npm ci" (clean install) command, this can be used if there are problems installing or compiling to also flush and recreate the node_modules folder.

To develop a Component or Service package can be added to the current solution as a project (instead of a zip).

Such a package needs to have '.sourcepath' file in the root that has 2 properties:

"srcDir": "projects/servoyextracomponents", "apiFile": "src/public-api"

srcDir is pointing to the angular main src dir (where the package.json is in of the package)

apiFile is pointing to the public-api.ts file inside that dir (without the ts)

Besides that pure build file a Component package the configuration are in the META-INF/manifest.mf

```
NG2-CSS-ClientLibs: ~@servoy/bootstrapcomponents/svy_bootstrapcomponents.css,
 ~@danielmoncada/angular-datetime-picker/assets/style/picker.min.css
NPM-PackageName: @servoy/bootstrapcomponents
NG2-Module: ServoyBootstrapComponentsModule
Entry-Point: dist/servoy/bootstrapcomponents
```

For a Service the spec file must have a section:
This one describes a service:

```
"ng2Config": {
"packageName": "@servoy/ngdesktopfile",
"serviceName": "NGDesktopFileService",
"entryPoint": "dist/servoy/ngdesktopfile"
},
```

The PackageName is the npm name.
The ServiceName is how the Service is exported like:

@Injectable()
export class NGDesktopFileService {

The entry point is the distribution folder what is the root of the npm package.

To debug you only need to run this:

npm run build_debug

in the [workspace]/.metadata/.plugins/com.servoy.eclipse.ngclient.ui/target folder.

this will then run in watch mode and will compile the NGClient2 together with all the binary (zip) a source (source projects) in 1 go.

For component packages and services we have a servoy public npm package @servoy/public - npm (npmjs.com)

That gives you the various api to talk to servoy or use helper functions/types: servoy-eclipse/public-api.ts at master · Servoy/servoy-eclipse (github.com)

Releasing a component

When releasing a component you need to build the component first

that means in the component root directory you have to do:

npm install (to make sure all package.json depedencies are there, node_modules folder is created, this should be done also for development)

npm run build (to really make the production build, compiling the component)

npm run make_release (this will generate then a zip file that is described in the scripts/build.js that should be adjusted to include all the stuff you want to have, that mostly the ng1 dirs/spec files and the "dist/xxx" folder)

this build and make_release scripts of npm are there when you generate a setup/project through the developer (the skeleton code)

else look at servoy-extra-components/package.json at master · Servoy/servoy-extra-components (github.com) (that make_release does a build and a package at one go)

and servoy-extra-components/build.js at master · Servoy/servoy-extra-components (github.com) for the script that makes the zip