Customizing SDK versions

Edit

It is easy for an app to customize which versions of the Windows SDK and WinUI 2.x to use.

Details

Each app has a file `ExperimentalFeatures.props` which describes the different SDK and library versions that the app depends on. This file can be found in the same directory as the app's `.sln` file.

Native Modules created in 0.64.3+ will locate and import this file at build time when they are linked into an app. This means that native modules will end up using the Windows SDK versions and WinUI 2.x versions that the app chose to use.

The `ExperimentalFeatures.props` file can be used to set a number of properties that determine which dependencies to use, including:

PROPERTY NAME	DESCRIPTION
`WinUI2xVersion`	Version of the WinUI 2.x package to use, e.g. `2.6.0`
`WindowsTargetPlatformVersion`	Version of the Windows platform SDK to use, e.g. `10.0.19041.0`
`UseHermes`	Whether to use the Hermes JavaScript engine.

Other properties

- the property is not reset after `ExperimentalFeatures.props` is included, or
- the property is only reset if it is empty (e.g. `Condition="'$(MyProperty)'=''"`)

Troubleshooting & debugging the build

You can gain insight into the state of a property across the build process by producing a binary build log (running a CLI build will do this), and opening the resulting `.binlog` file in the MSBuild Structured Log Viewer. es and targets

Updating your app

Using a dependency involves the following two tasks:

- Referencing the appropriate NuGet package so it gets downloaded
- Importing the package's build properties and targets

For C# apps, the two steps are combined thanks to the built-in `<PackageReference>` support. However, C# apps restore NuGet packages to a location under your user profile, whereas C++ projects expect NuGet packages to be in the solution directory. The `Microsoft.ReactNative` project which implements the RNW framework is a C++ project so it needs to be able to find the WinUI package under the solution `packages` directory. A workaround for this, is to edit the file `node_modules\react-native-windows\Microsoft.ReactNative\packages.config` to update the WinUI version and have it restore to the solution directory correctly.

C++ apps use `packages.config` to specify the set of NuGet packages, and then manually import the right `.props` and `.targets` files from the package.

When you create a C# or C++ React Native for Windows app, it is written in such a way that the WinUI version it uses is parametrized (i.e. it depends on the value of the

the updated package version.

windows\ExperimentalFlags.props

Copy

```
<PropertyGroup>
  <!-- other properties -->
  <WinUI2xVersion>2.6.0</WinUI2xPackageVersion>
</PropertyGroup>
```

‹ Developing Windows apps on a non-Windows PC

Managing C++ dependencies ›