Fetch Optional (Internal) Tooling

This is an implementation plan for how to fetch sensitive internal tools during a .NET Core build.

Uploading a new tool

The tool is put in a NuGet package and uploaded to a VSTS feed. VSTS feeds require authentication for any operation, and are secure.

Fetching during the build

To fetch internal tooling in your local dev build, see the Running CoreFx tests on UAP (CoreCLR scenario) OneNote page

An `optional-tool-runtime/project.json` file in BuildTools specifies all required tooling that is only available from the internal VSTS feed. This is similar to `tool-runtime/project.json`.

In the official build, three properties are set for the `sync` call:

```
OptionalToolSource=https://devdiv.pkgs.visualstudio.com/_packaging/dotnet-core-internal-tool
OptionalToolSourceUser=dn-bot
OptionalToolSourcePassword=******
```

A target in BuildTools runs before the main project package restore, detects that these properties are set, then restores `optional-tool-runtime/project.json` into the `packages` directory. Build steps that need an optional tool can find it using `PrereleaseResolveNuGetPackageAssets`.

The path to the project file can be overridden to specify repo-specific tooling, like in CoreFX: dir.props#L303, optional.json.

Devs who have the optional tooling packages but don't have convenient access to the VSTS feed can set `OptionalToolSource` to a directory to use it as an optional tool package feed.

If `OptionalToolSource` isn't set, no optional tooling is restored.

Was this helpful? 👍 👎