CMPS 3680 Homework 10

Fall 2024

Application Deployment

Deployment in application development is the process of delivering a software application to its intended environment, such as a production server, where end-users can access and interact with it. This phase involves moving the application from development or testing environments to a live environment and ensuring it runs smoothly and meets user expectations.

Key aspects of deployment include preparing the application for release, configuring the production environment, and managing dependencies such as databases, APIs, and servers.

Deployment Tool Examples

Java

- **jpackage**: Packages Java applications into platform-specific installers (e.g., `.exe`, `.dmg`, `.deb`).
- **Spring Boot Maven/Gradle Plugins**: Creates executable JARs or WARs, often used in web application deployments.
- **Jenkins Pipelines**: Automates deployment tasks, integrating with Java build tools.

Python

- **PyInstaller**: Packages Python programs into standalone executables for distribution.
- **Flask/FastAPI Deployment Tools**: Often paired with WSGI servers like **Gunicorn** or **uWSGI** for web apps.

JavaScript/Node.js

- **pkg**: Creates standalone executables from Node.js applications.
- **Webpack**: Optimizes and bundles JavaScript applications for deployment.
- **Vercel**: Simplify deployment of frontend frameworks like React or Next.js.
- **Electron**: embeds Chromium and Node.js to enable web developers to create desktop applications.

C#/.NET

- **MSIX Packaging Tool**: For creating modern Windows app packages.
- **Visual Studio Publish Tool**: Facilitates the deployment of .NET applications to servers, Azure, or containers.
- **WIX Toolset**: set of build tools that build Windows Installer packages using compiled source code

C++

- **CMake/CPack**: Used to build and package C++ applications into platform-specific installers.
- **NSIS (Nullsoft Scriptable Install System)**: Creates Windows installers for native C++ applications.

Go (Golang)

- **goreleaser**: Automates the build and release of Go applications, including binary packaging for multiple platforms.
- **Go Build**: Native command for creating platform-specific binaries.

Rust

- **Cargo Bundle**: Builds and bundles Rust applications into native executables or installers.
- **Cross**: Cross-compiles Rust binaries for various platforms.

Dependency Injection

"Dependency injection" can seem intimidating at first, but it's really just a fancy name for a programming technique you are probably already using. Here's the basic idea: Instead of declaring/initializing an object inside of another object or function, you pass an already initialized object instead. This creates a "loose coupling" that allows the object being passed to be accessed, manipulated, or reused.

A perfect example of this that we have already seen with the View/Controller relationship:

- One view could potentially be modified by multiple/different controllers
- One controller could modify multiple/different views
- Thus, it does **not** make sense to declare/initialize a view INSIDE of a controller (strong coupling)
- It is better to create views OUTSIDE of controllers, and pass them to controllers as needed (loose coupling)

Another use case for dependency injection

- You have a "User" class that requires authentication
- You want to allow multiple types of authentication, but only one per user
- Therefor, it wouldn't make sense to declare/initialize the authentication object inside the user class
- It would make more sense to have a "Auth" interface that can have multiple implementations
- Then, when you create the user, you simple pass the already initialized "Auth" object to the user object

Additional Resources

- NPM Trends for Javascript Deployment
- "A 25-dollar term for a 5-cent concept" (jamesshore.com)
- Dependency Injection, The Best Pattern (CodeAesthetic)