Chapter 2: Software List

This chapter explores the actual tools we are using in our example, the basic understanding needed for each tool, and an explanation of why we're doing things this way.

In short, our example relies on:

- **Git**
  - A **Git** frontend if you are unused to Git - **gitkraken** is nice for Windows
- **Bitbucket**
- A good text editor (**VSCode** is fine, I prefer Sublime Text)
- The SF command line
  - **SFDMU**, a SF command line extension
  - **SGD**, a SF command line extension
- **JIRA**

You can completely use other tools if your project, your client, or your leadership want you do use other things.
The main reason we are using these in this example is that it relies on a tech stack that is very present with customers and widely used at a global level, while also leveraging reusable things as much as possible - technically speaking a lot of the configuration we do here is directly reusable in another pipeline provider, and the link to tickets is also something that can be integrated using another provider.

In short "use this, or something else if you know what you're doing".

So What are we using

The CLI

The first entrypoint into the pipeline is going to be the **Salesforce Command Line.**You can download it here.
If you want a graphical user interface, you should set up VSCode, which you can do by following this Trailhead. You can start using the CLI directly via the terminal if you already know what you're doing otherwise. If you're using VSCode, download Azul as well to avoid errors down the line.

> We'll be using the Salesforce CLI to:
> - login to organizations, and avoid that pesky MFA;
> - pull changes from an organization once our config is done;
> - rarely, push hotfixes to a UAT org.

For some roles, mainly architects and developers, we will also use it to:

- validate deploys to specific orgs in cases of hotfixes;
- setup SGD jobs in cases of commit-based deploys or destructive changes;
- setup SFDMU jobs for any data-based transfers.

What this actually does is allow you to interact with Salesforce. We will use it to get the configuration, security, and setting files that we will then deploy.

> This allows us not only to deploy, but also to have a backup of the configuration, and an easy way to edit it via text edition software.

The configuration needed is literally just the installation to start - we'll set up a full project later down the line.

GIT

You'll then need to download Git, as well as a GUI if you're not used to using it directly from the command line. Git is VERY powerful but also quite annoying to learn fully, which is why we will keep its usage simple in our case.

> We'll be using Git to:
> - version our work so we can easily go back to earlier configurations in case of issues;
> - document what we did when we modified something;
> - get the work that other people have done;
> - upload our work to the repositories for the project.

You'll need a bit more configuration once you're done installing - depending on the GUI you use (or if you're using the command line) the *how* depends on the exact software, but in short you'll need to configure git with your user name and your user email.

> Logging in to Bitbucket and getting your repository from there will come later - once you've given your username and email, and configured your UI, we will consider that you are done for now.

If you're a normal user, this is all you'll see of git.
If you're a Dev or an Architect, you'll also be using the Branches and Merges functions of Git - mostly through the Bitbucket interface (and as such, with Pull Requests instead of Merges).

Bitbucket

As said in intro, we're using bitbucket because we're using bitbucket. You can use Gitlab, Gitea, whatever - but this guide is for bitbucket. If you're using GitHub... why aren't you using DevOps Center ?

Bitbucket, much like Salesforce, is a cloud solution. It is part of the Atlassian cloud offering, which also hosts JIRA, which we'll be configuring as well. You'll need to authenticate to your workspace (maybe get your Administrator to get you logins), in the format https://bitbucket.org/myworkspace

You will see that Bitbucket is a Git Server that contains Git Repositories.
In short, it is the central place where we'll host the different project repositories that we are going to use.
Built on top of the Git server are also subordinate functions such as Pull Requests, Deployments, Pipelines - which we're all going to use.

Seeing as we want this to be connected with our Atlassian cloud, we'll also ask you to go to https://bitbucket.org/account/settings/app-passwords/ which allows you to create application passwords, and to create one for Git.

In detail:

- **Repositories:** Developers store their Salesforce metadata and code in Bitbucket repositories. Each repository can represent a project or a component of a larger Salesforce application.
- **Branching:** Developers create branches for new features, bug fixes, or enhancements. This allows multiple developers to work on different parts of the codebase simultaneously without interfering with each other.
- **Pull Requests:** When a feature or bug fix is complete, a pull request is created. Other team members review the changes before they are merged into the main branch, ensuring code quality and consistency.
- **Commits:** Developers commit their changes to Bitbucket, providing a detailed commit message. These messages often include references to JIRA ticket numbers (e.g., "Fixed bug in login flow [JIRA-123]").
- **JIRA:** When a commit message includes a JIRA ticket number, JIRA can automatically update the status of the ticket, link the commit to the ticket, and provide traceability from issue identification to resolution.
- **Pipelines:** Bitbucket Pipelines can be configured to automatically build, test, and deploy Salesforce code changes. This ensures that changes are validated before being merged

and deployed to production. It does so using **Deployments** - which in bitbucket means "the installation of code on a remote server", in our case Salesforce.

Extra Stuff

SGD

SGD, or Salesforce-Git-Delta is a command line plugin that allows the CLI to automatically generate a package.xml and a destructivechanges.xml based on the difference between two commits.
It allows you to do in Git what the CLI does alone using Source Tracking.
Why is it useful then ? Because Source Tracking is sometimes buggy, and also because in this case we're using Bitbucket, so it makes generating these deployment files independent from our machines.
SGD is very useful for inter-org deployment, which should technically be quite rare.

SFDMU

SFDMU, or the Salesforce Data Move Utility, is another command line plugin which is dataloader on steroids for when you want to migrate data between orgs or back stuff up to CSVs.
We use this because it allows migrating test data or config data (that last one should be VERY rare what with the presence of CMTD now) very easily including if you have hierarchies (Contacts of Accounts, etc).

Revision #9
Created 19 June 2024 15:14:53 by Windyo
Updated 4 July 2024 15:44:16 by Windyo