42 command line prompts every SAP BTP developer needs to know

Essential for SAP Developers getting acquainted with and mastering BTP development and administration

By Ryan Fleischmann - SAP BTP Architect, Developer

Introduction

One of the big changes every SAP ABAP or SAP-focused integration developer goes through in adopting BTP is the command line. While many SAP developers know a number of keyboard shortcuts and transaction codes for SAP GUI and other SAP developer tools on-prem - they may not have always gotten good practice with using the kind of command-line prompts typically used in DevOps-focused/Cloud-Native development.

The command line can be intimidating for developers not accustomed to using it regularly.
Though Integration Suite, SAP Business App Studio and SAP Build Code may very often feature UI screens for these same tasks - the reality of devops-focused/cloud native development is that the command line is necessary for many development, tech administration, and other troubleshooting tasks.

But how and where do you start? For many SAP BTP developers, basic Linux and git commands are to be your most commonly used CLI commands. If you don't know those - you'll frankly be helpless with everything else related to CLIs.

From there - many of your cloud native developments on BTP will likely be Node.js applications deployed to the Cloud Foundry environment. In BTP you do have other options and considerations too for many other programming languages (Python, Golang), SDKs (specifically for AWS, GCP, Azure), DevOps/Containerization/IaaS+IaC tools (Kyma/Kubernetes, Docker, Terraform), and other command line tools. The abundance of advanced tools and options is nice in many ways - but terrible in another: *what do I really need? Are there some basics worth considering first?*

To that end: provided here is a list of 42 frequently used command line prompts belonging to what I consider essential (or at *least* very highly valuable) to know for successfully contributing to a SAP BTP project at a technical level. Moreover - a great many of these are helpful for application development and integration in SAP environments still to adopt BTP (including ECC ones!) as well.

General Linux (9)

–

`cd <<directory_path>>`

Changes the directory that the terminal is making changes to. Use `cd ..` to go one directory level up

–

`ls`

Lists the contents of the present directory

–

`mkdir <<new_directory>>`

Creates a new directory with the specified name

–

`touch <<filename.txt>>`

Creates the specified file

–

`nano <<filename.txt>>`

Opens the text editor to edit the specified file.

–

`echo “hello world”`

Basic “hello world” in linux shell scripting.

-

`sudo`

Elevates the command line to super user privileges - typically for the purpose installing additional CLI tools. Use with care!

–

Liked this guide?

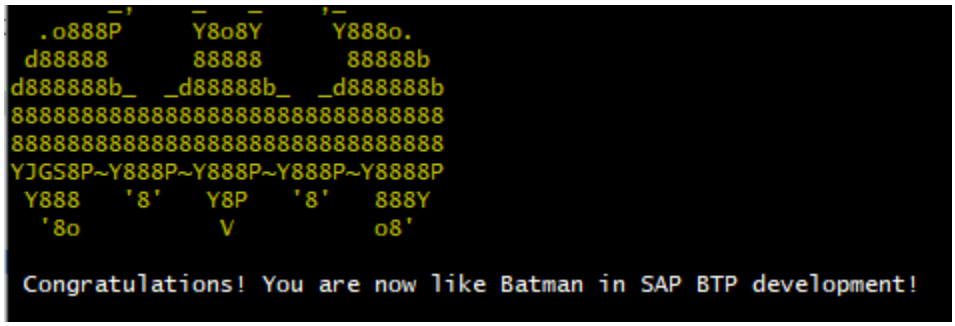
Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

`sh <<bashscript.sh>>`

Executes a bash script as specified

You know you can write your own custom bash scripts to automate tasks? I created an example here that I like to call “the Batman test of SAP dev readiness”. Check it out here: https://github.com/fleischr/sapbtp_dev_readiness

It supplements this document really well by helping you check for which CLI tools you’ve installed in your environment. When it detects you got everything set up - it shows you this bat signal! Pretty neat, huh?



-

`curl <<url>>`

This command executes HTTP requests. This can be either for testing connectivity across systems or creating data and configuration in systems. Lookup curl -h for advanced settings to pass JSON or XML payloads, add additional HTTP headers and others.

Liked this guide?

Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

Git (10)

–

`git init`

Run this to set up your directory as a git repository

–

`git clone <<github.com/your_org/your_repo>>`

Copies the github repo to your local workspace

–

`git branch <<branch>>`

Creates a new branch with the given name

–

`git checkout <<branch>>`

Switches the github branch your code editor is currently using. Use this either when first creating new branches for development

–

`git add <<filename>>`

Adds the specified file to the git staging area. Use `git add .` to automatically stage all changed files

–

`git rm <<filename>>`

Deletes a file from the local directory, removes it from git tracking at the same time

-

```
git commit -m “<<commit message>>”
```

Organizes the changes in the staging area under one commit. Think of this a lot like having a fully released CTS and tasks

–

```
git push origin <<branch>>
```

Pushes all the latest commits to the remote repository for the given branc

–

```
git pull origin <<branch>>
```

Retrieves the latest state of the branch from the remote repo. Use this to sync your project to an earlier state or to preview changes created by your colleagues.

–

```
git remote origin set-url
https://<<githubuser>:<<token>>@github.com/<<org>/<<repo>>
```

Updates the authentication and/or url used for the remote repository. Use this either when your access tokens expire or you need to maintain the code in new, unforked git repository

Liked this guide?

Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

Other General CLI conventions (3)

`*** >> <<filename.txt>>`

This is moreso an addition to a command than a command itself.

But a very powerful thing that it does is take the command output and saves it to a file!

Example echo `"# new readme" >> readme.md` creates a new readme file with the header
–

`*** -h or --help`

Any CLI tool worth using will include options to request more information about what commands are available to use. If you're figuring out any CLI tool for the first time try exploring the help a bit more before googling for a solution.

`*** version / -version`

Checking the CLI version is important to do from time to time!

Node.js/NPM (5)

–

`node <<script.js>>`

Runs the specified node.js script. Sorta like SE38 - but for javascript.
–

`npm init`

Bootstraps a new node.js project. Use `npm init -y` to speedrun the walkthrough process.
–

`npm install <<name_npm_module>>`

Also commonly abbreviated to `npm i <<name_npm_module>>`

Liked this guide?
Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

Downloads and installs the specified Node.js module from the NPM registry. The name of the module dependency and requirements is included in package.json

Additional CLI tools - such the the CAP CDS cli tools are commonly installed to operating systems more generally through NPM using –global or -g. Adding further CLI tools into your repertoire supercharges your developer capabilities!

```
npm install --global @Sap/cds-dk

npm i -g abaplint
```

–

`npm run <<script_mapping_in_package.json>>`

Runs the the command mapped in the “scripts” section of package.json

–

`npm update`

Updates the installed node modules in the directory to their latest supported version

UI5/Fiori(4)

Hint: add these commands to your environment by first running
npm i -g yeoman-generator and

`yo`

Begins the Yeoman generator to create a new Fiori app

`ui5 init`

Configures the SAP Fiori Tools CLI for the given UI5 project

`ui5 add`

Adds SAPUI5/OpenUI5 modules to the project. Automates adding the necessary declaration in Component.js and elsewhere

Liked this guide?
Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

`ui5 serve`

Starts a local UI5 server for development. This is particularly useful for developing the SAPUI5 app without needing to deploy to the ABAP development system

Cloud Foundry / SAP BTP (6)

–

`mbt build`

Based upon configuration in a given mta.yaml - generates a specified *.mtar file in the mta_archive directory. The *.mtar file is then used to deploy the application in a later step.

–

`cf login`

Logs into the cloud foundry instance. Additional prompts follow to specify API endpoint, credentials, org, and space.
You can specify all these at once with this format

```
cf login -a <<API_URL>> -o <<CF_ORG>> -p "<<PASSWORD>>" -s <<SPACE>>
```

Alternatively, check the help for SSO based logins.

Also note that 1.) some special characters may present issues when entered directly through the CLI - so always wrap the password in double quotes. And 2.) your S-user password may actually differ from your SAP universal id password! If possible - use SSO instead for an easier login experience. I recommend SSO particularly when working from BAS or VS Code.

–

`cf deploy <<you_archive.mtar>>`

Uploads the contents of the mtar and executes deployment processes for each component of the cloud foundry app. The app will be deployed to the Cloud Foundry org and space specified at login

–

`cf env <<yourAppModule>>`

At a certain point in your development lifecycle - you'll likely need to get deep in the weeds of how your application is consuming other service instances on the BTP subaccount. To do this - You'll also want your development IDE to use service bindings as if your application BAS or VS Code were the subaccount itself. But how do you do that? And how do you get that information?

In BAS or VS code - the default-env.json is your tool to accomplish local debugging with remote service instances. Running the `cf env` command will help you produce the necessary data for VCAP_SERVICES and VCAP_APPLICATION sections of JSON of default-env.json.

Default-env.json may contain sensitive credentials data - not appropriate for version control. Further the service bindings automatically rotate on deployments. That said - an active BTP developer should be comfortable using the cf env command to gather the needed environment data to develop and troubleshoot applications!

–

`btp login --sso`

Logs into the SAP BTP subaccounts using SSO session from your browser.

–

`btp target`

Sets the which global accounts and subaccounts you are working upon in the CLI

–

CAP/CDS (4)

–

`cds init`

Bootstraps the basic structure of the CAP project

–

`cds add <<module>>`

Adds CAP-specific modules - typically needed for use and deployment to SAP BTP. Some examples include `cds add mta` for builds, `cds add xsuaa` for authentication, `cds add hana` for use of a HANA database instance.

Liked this guide?

Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch

You can also use `cds add sample` to generate a full CAP application example. Or use `cds add tiny-sample` for an easier to use starter project.

–

`cds import <<apireference.json>>`

Creates an interface to an API using an OpenAPI definition or oData EDMX file

–

`cds watch`

Starts up the CAP application. You can use the addition `--with-mocks` to specify your application to use mocked data from CSV files for development and testing purposes

–

ABAP (1)

`abaplint`

Ready to have your mind blown? You can run ABAP code quality checks using Node.js and abapGit! This makes it especially useful for creating CI/CD pipelines for many SAP projects regardless of their cloud maturity and progress with BTP adoption. You can use the abaplint tool offline from BTP and other cloud systems via the command line too. And given abapGit's extended support even to ECC systems - this means you could even use abaplint to automate your ECC code quality scans as well. That sounds like the perfect way to keep your CLI scripting skills sharp while you wait to see your organization's BTP plans heat up.

I do also recommend you also check out the commercially supported version (see abaplint.app) to support the developers great open source work for SAP developers and to learn more on how to master the tool.

Appendix

https://git-scm.com/downloads

https://nodejs.org/en/download/source-code

https://github.com/fleischr/sapbtp_dev_readiness

https://sap.github.io/ui5-tooling/v3/

https://www.npmjs.com/package/@ui5/cli

https://github.com/cloudfoundry/cli/wiki/V8-CLI-Installation-Guide

https://help.sap.com/docs/btp/sap-business-technology-platform/download-and-start-using-btp-cli-client

https://www.npmjs.com/package/@sap/cds-dk

https://www.npmjs.com/package/@abaplint/cli

https://abaplint.app/

Liked this guide?

Feel free to show your appreciation here!
https://buymeacoffee.com/ryfleisch