- 1. Tools
 - o 1.1. iTerm2
 - o 1.2. Homebrew
 - o 1.3. K9S
 - 1.4. Xcode (OPTIONAL not initially needed)
- 2. CLI enhancements
 - o 2.1. Framework
 - o 2.2. P10k
 - o 2.3. Plugins
 - 2.3.1. Kubectl
 - 2.3.2. zsh-autosuggestions
 - 2.3.3. zsh-syntax-highlighting
 - o 2.4. Other enhancements
 - 2.4.1. kubecolor
 - 2.4.2. Autojump
 - 2.4.3. Exa
 - 2.4.4. Context and Namespace alias

1. Tools

1.1. iTerm2

I guess you are already using iTerm2. If you are not, start by installing it iterm2.com and use that instead of terminal.

1.2. Homebrew

Also another tool that we will need in the journey is homebrew (brew.sh). If you don't already have it, just install it by:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

after the installer is done, you need to execute a couple of lines to add it to your path (*replace with your current username* in the first one)

```
(echo; echo 'eval "$(/usr/local/bin/brew shellenv)"') >>
/Users/szigeti/.zprofile
eval "$(/usr/local/bin/brew shellenv)"
```

1.3. K9S

The best CLI kubernetes explorer. Install it also with Homebrew:

```
brew install derailed/k9s/k9s
```

Documentation is here

1.4. Xcode (OPTIONAL - not initially needed)

A third tool to keep in consideration are Apple Xcode tools. Do:

```
sudo rm -rf /Library/Developer/CommandLineTools
sudo xcode-select --install
```

...and accept the license pop-up.

2. CLI enhancements

2.1. Framework

Next thing to consider is the installation of a framework. Here we chose OhMyZsh. Execute the following:

```
sh -c "$(curl -fsSL
https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)
"
```

```
Last login: Tue Sep 26 11:22:07 on console
twomeTypers-Mac-mini-3 ~ X sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"
Cloning Oh My Zsh...
remote: Enumerating objects: 1361, done.
remote: Compressing objects: 100% (1361/1361), done.
remote: Compressing objects: 100% (1361/1361), 2.00 MiB I 8.95 MiB/s, done.
Receiving objects: 100% (1361/1361), 2.00 MiB I 8.95 MiB/s, done.
Resolving deltas: 100% (33/33), done.
From https://github.com/ohmyzsh/ohmyzsh
* [new branch] master -> origin/master
Branch master' set up to track remote branch 'master' from 'origin'.
Already on 'master'
JUSers/tvan
Looking for an existing zsh config...
Found /Users/ivan/.zshrc. Backing up to /Users/ivan/.zshrc.pre-oh-my-zsh
Using the Oh My Zsh template file and adding it to /Users/ivan/.zshrc.

### Follow us on Twitter: @ohmyzsh

* John our Discord community: Discord server

### Follow us on Twitter: @ohmyzsh

* Join our Discord community: Discord server

### Follow us on Twitter: @ohmyzsh

* Join our Discord community: Discord server

### Follow us on Twitter: @ohmyzsh

* Join our Discord community: Discord server

#### ### Get stickers, t-shirts, coffee mugs and more: Planet Argon Shop
```

2.2. P10k

Now let's add some customization to the prompt with powerlevel10k P10k. If you just run:

```
brew install powerlevel10k

echo "source $(brew --prefix)/share/powerlevel10k/powerlevel10k.zsh-
theme" >>~/.zshrc
```

... and then close the terminal window and open a new one, and follow the on screen menu for customizing the prompt to match your likes. It will be important later on, to show the cluster and namespace in which you are working. You can always use defaults if you don't have a personal preference on something. Usually first line is the typical choice.

```
ivan@Ivans-Mac-mini-3:~

This is Powerlevel10k configuration wizard. You are seeing it because you haven't defined any Powerlevel10k configuration options. It will ask you a few questions and configure your prompt.

Install Meslo Nerd Font?

(y) Yes (recommended).

(n) No. Use the current font.

(q) Quit and do nothing.

Choice [ynq]:
```

2.3. Plugins

Lets add some plugins to make typing easier. They are all here, but there is a minimum set recommended.

You can enable the ones you need by editing the ~/ . zshrc file:

```
vim ~/.zshrc
```

Locate the plugin section, usually like this:

```
76 # Which plugins would you like to load? 20
77 # Standard plugins can be found in $ZSH/pl
78 # Custom plugins may be added to $ZSH_CUST
79 # Example format: plugins=(rails git textm
80 # Add wisely, as too many plugins slow dow
81 plugins=(git)
```

These are the ones I've enabled:

```
copybuffer
)
```

Most of them are already inside the plugin directory of your Oh-My-Zsh (\$HOME/.oh-my-zsh/plugins/), but some of them need to be downloaded. I'll detail them bellow in a particular section for each.

Also, to make the autocompletion works correctly while using the Oh-My-Zsh framework, you need to add the following PATH modification before sourcing the Oh-My-Zsh, as per here

```
FPATH="$(brew --prefix)/share/zsh/site-functions:${FPATH}"
```

```
copyfile
copybuffer
frame properties
fra
```

As a reference, this is my full ~/.zshrc attached

2.3.1. Kubectl

Kubectl autocompletion. This plugin is already downloaded. You just need to enable it on the ~/.zshrc

It gives you a nice set of alias that you can use to shorten your calls to 'kubectl' commands:

kaf ... -> kubectl apply -f ... kgpo -> kubectl get pods

... all the available aliases are listed here

2.3.2. zsh-autosuggestions

You need to download this plugin and also declare it in the plugin section of the ~/.zshrc. Download it with:

```
git clone https://github.com/zsh-users/zsh-autosuggestions
${ZSH_CUSTOM:-~/.oh-my-zsh/custom}/plugins/zsh-autosuggestions
```

2.3.3. zsh-syntax-highlighting

Declare the plugin on your ~. /zshrc and exec this also to download it:

```
git clone https://github.com/zsh-users/zsh-syntax-highlighting.git
${ZSH_CUSTOM:-~/.oh-my-zsh/custom}/plugins/zsh-syntax-highlighting
```

2.4. Other enhancements

2.4.1. kubecolor

Install this tool with homebrew:

```
brew install hidetatz/tap/kubecolor
```

and add this at the end of your .zshrc

```
# get zsh complete kubectl
source <(kubectl completion zsh)
alias kubectl=kubecolor
# make completion work with kubecolor
compdef kubecolor=kubectl</pre>
```

I have several alias in my ~/.zshrc that make my life easier:

```
148

149 alias ls="exa -1 --classify --group-directories-first"

150 alias ll="exa --long --header --classify --icons --group-directories-first --no-permissions"

151 alias lp="exa --long --header --classify --icons --group-directories-first"

152 alias la="exa -a --long --header --classify --icons --group-directories-first"

153 alias lt="exa --tree --long --header --classify --icons --group-directories-first --level=3"

154 alias t="exa --tree --header --classify --icons --group-directories-first --level=3"

155 alias kotx="kubectx"

156 alias kns="kubectx"
```

2.4.2. Autojump

Very useful tool, that allows you to jump directly into the directory of your project just using its name.

```
brew install autojump
```

That also install a wrapper j, so I do very often things like 'j thisProject'

2.4.3. Exa

Good replacement for 'ls', you can see the alias in the previous picture in my ~/.zshrc. The tool can be found here, but you can install it directly with homebrew with:

```
brew install exa
```

See some examples of the use here. Actually when you add some alias, you don't have to do anything special for using it, it is called everytime you do an 'ls'. These are my alias for it on my ~/*zshrc file:

```
la='exa -a --long --header --classify --icons --group-directories-first'
ll='exa --long --header --classify --icons --group-directories-first --
no-permissions'
lp='exa --long --header --classify --icons --group-directories-first'
ls='exa -1 --classify --group-directories-first'
lt='exa --tree --long --header --classify --icons --group-directories-
first --level=3'
t='exa --tree --header --classify --icons --group-directories-first --
level=3'
```

2.4.4. Context and Namespace alias

These two alias can also be added to the ~/.zshrc to allow easy change of cluster/namespace:

```
alias kx='f() { [ "$1" ] && kubectl config use-context $1 || kubectl
config current-context ; } ; f'

alias kn='f() { [ "$1" ] && kubectl config set-context --current --
namespace $1 || kubectl config view --minify | grep namespace | cut -d" "
-f6 ; } ; f'
```

But you can also install 'kubectx' and 'kubens' tools described here, which will do the same but also will display a list of the available choices for context and namespace:

```
brew install kubectx
```