Windows shortcuts

@PrasoonPratham

Win + D
Show Desktop

Win + PRTSCN
Print Screen

Win + (-/+)
Zoom in/out

Win + Shift + Up/Down/Left/Right
Dock window up/down/left/right of the screen

Alt + Tab
Switch between Apps

Crtl + Esc
Open Start Menu

Win + Tab
Open Task View

Win + Crtl + D
Create new virtual desktop

Shift + Delete
Permanently delete app without moving to Bin

Crtl + E
Open File Explorer

Crtl + W
Close Window

9 MacOS Shortcuts

~ @PrasoonPratham

Command-left/right arrows
Change volume in small increments

Command + Comma
open the application's Preferences

Command + Space
open spotlight

Command+Option+D
Show Dock

Fn+left/right arrow
Move to end of page

Control + Command + Space
Open Emoji picker

Option + Command + Esc
Force close App

Option + Shift + Volume
Change volume in small increments

Command + W
Closes the active window

Terminal Commands

`mkdir [name]`
Make directory

`cd ..`
Go up one step

`ls`
List files in current directory

`pwd`
Show current directory

`cd`
Go to root directory

`touch [fileName]`
Make a file

`cat [fileName]`
Read a file

`rm -R [/directory]`
Remove Folder

`rm [/file]`
Remove File

`mv [filename] [path/to/new/file/location]`
Move file to new location

Hello World in 30 different languages

C

```c
#include

int main(void)
{
    puts("Hello, world!");
}
```

Matlab

```
disp('Hello, world!')
```

Pascal

```
WriteLn('Hello, world!');
```

Go

```
println('Hello, world!');
```

F#

```
printfn "Hello World"
```

Lisp

```
(print "Hello world")
```

C#

```
Console.WriteLine("Hello, world!");
```

Ruby

```
puts "Hello World!"
```

Java

```
System.out.println("Hello World!");
```

JavaScript

```
console.log 'Hello, world!'
```

C++

```
#include

int main()
{
    std::cout << "Hello, world!
";
    return 0;
}
```

CoffeeScript

```
console.log 'Hello, world!'
```

Python

```
print('Hello, world!')
```

PHP

```
echo "Hello World!";
```

Algol

```
BEGIN DISPLAY("HELLO WORLD!") END.
```

Delphi

```
program HelloWorld;
begin
  Writeln('Hello, world!');
end.
```

Assembly

```
    global  _main
    extern  _printf

    section .text
_main:
    push    message
    call    _printf
    add     esp, 4
    ret
message:
    db  'Hello, World', 10, 0
```

Pascal

```
program HelloWorld(output);
begin
  Write('Hello, world!')
end.
```

Perl

```
print "Hello, World!\n";
```

Tcl

```
puts "Hello World!"
```

Cobol

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. hello-world.
       PROCEDURE DIVISION.
           DISPLAY "Hello, world!"
           .
```

Dart

```
main() {
  print('Hello World!');
}
```

Kotlin

```
fun main(args: Array<String>) {
    println("Hello World!")
}
```

TypeScript

```
console.log 'Hello, world!'
```

Scala

```
object HelloWorld extends App {
   println("Hello, World!")
 }
```

Haskell

```
module Main where

main :: IO ()
main = putStrLn "Hello, World!"
```

R

```
cat("Hello world\n")
```

Swift

```
println('Hello, world!');
```

HTML

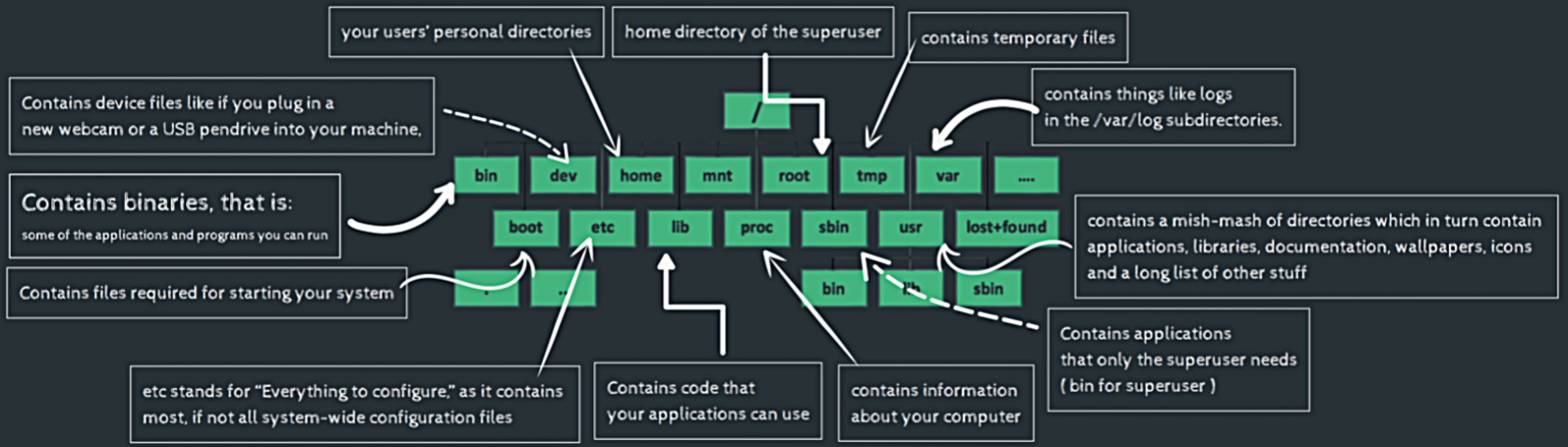
```
Hello world
```

Fortran

```
program helloworld
      print *, "Hello world!"
end program helloworld
```

Linux File System

~ @PrathamPrasoon



Python 3.9 Must Know Features

~ @PrasoonPratham

Merge Dictionaries

```
> a = {"key1":1,"key2":2}
> b = {"key3":3,"key4":4}
> print(a | b)
> {"key1":1,"key2":2,"key3":3,"key4":4}
```

Update Dictionaries

```
> a = {"key1":1,"key2":2}
> b = {"key2":3,"key4":4}
> print(a |= b)
> {"key1":1,"key2":3,"key4":4}
```

Support For IANA timezone In DateTime

```
from zoneinfo import ZoneInfo
from datetime import datetime

dt = datetime(2000, 01, 25, 01,
     tzinfo=ZoneInfo("America/Los_Angeles"))
```

Random Byte generation

```
random.Random.randbytes()
```

Remove Prefix/Suffix

```
> string = "Hello World"
> string.removeprefix("Hello ")
World
> string = "Hello World"
> string.removesuffix("World")
Hello
```

Consistent Package Import Errors

The __import__() now raises ImportError instead of ValueError.

Ability To Cancel Concurrent Futures

A new parameter cancel_futures has been added to the concurrent.futures.Executor.shutdown().

New Parser, more flexible than the previous version.

Type Hinting For Built-in Generic Types

```
def print_value(input: str):
  print(input)
# We will get notified if the input is not a string
```

OpenCV Cheatsheet

```
//Blurring an image
import cv2
blurred = cv2.GaussianBlur(image, (51, 51), 0)
viewImage(blurred, "Blurred doggo")
```

```
//Importing an Image & Viewing it
import cv2
image = cv2.imread("./Path/To/Image.extension")
cv2.imshow("Image", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

```
//Cropping an image
import cv2
image = cv2.imread("./Path/To/Image.extension")
cropped = image[10:500, 500:2000]
viewImage(cropped, "Image after cropping.")
```

```
//Rotating an Image
import cv2
(h, w, d) = image.shape
center = (w // 2, h // 2)
M = cv2.getRotationMatrix2D(center, 180, 1.0)
rotated = cv2.warpAffine(image, M, (w, h))
viewImage(rotated, "image rotated by 190 degrees")
```

```
//Resizing an image
import cv2
scale_percent = 20
width = int(img.shape[1] * scale_percent / 100)
height = int(img.shape[0] * scale_percent / 100)
dim = (width, height)
resized = cv2.resize(img, dim,
interpolation = cv2.INTER_AREA)
viewImage(resized, "After resizing with 20%")
```

```
//Writing on an image

import cv2
output = image.copy()
cv2.putText(output, "Text", (1500, 3600),
cv2.FONT_HERSHEY_SIMPLEX, 15, (30, 105, 210),
 40)
viewImage(output, "image with text")
```

```
//Saving the image
import cv2
image = cv2.imread("./Import/path.extension")
cv2.imwrite("./Export/Path.extension", image)
```

```
//Drawing a Rectangle in the image
import cv2
output = image.copy()
cv2.rectangle(output, (2600, 800), (4100, 2400),
 (0, 255, 255), 10)
viewImage(output, "Image with rectangle")
```

PrasoonPratham

```
//Grayscale effect
import cv2
image = cv2.imread("./Path/To/Image.extension")
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
ret, threshold_image = cv2.threshold(im, 127, 255, 0)
viewImage(gray_image, "Gray-scale image")
viewImage(threshold_image, "Black & White image")
```

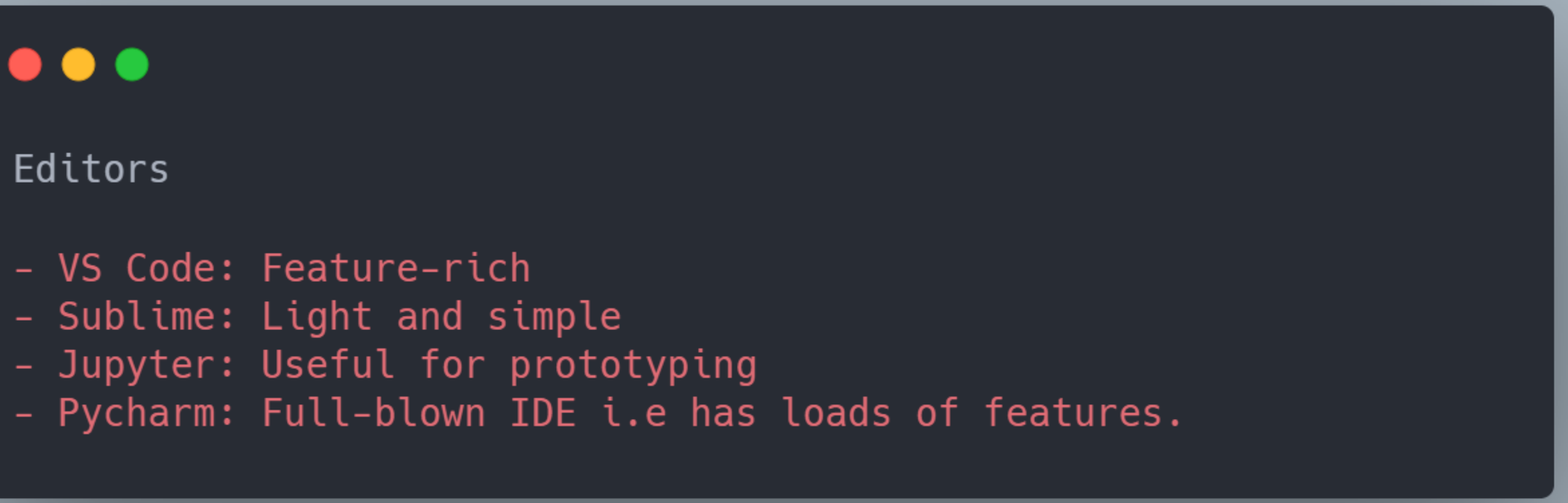
assertions.py

```python
#This function returns the average of numbers in a list

def avg(marks):
    assert len(marks) != 0,"List is empty."
    # This assert statement throws an error when the list is empty
    # assert <condition>,<error message>
    return sum(marks)/len(marks)

mark2 = [55,88,78,90,79]
print("Average of mark2:",avg(mark2))
#Output: Average of mark2: 78.0

mark1 = []
print("Average of mark1:",avg(mark1))
#Output: AssertionError: List is empty.
#The assert statement that we used earlier throws an error here
```


Editors
- VS Code: Feature-rich
- Sublime: Light and simple
- Jupyter: Useful for prototyping
- Pycharm: Full-blown IDE i.e has loads of features.

Python Projects

- Build a Website with Django/Flask
- Use a WebScraper
- Create a Game with PyGame
- Build a GUI with Tkinter/PyQt5
- Robotics with python

Basics
- Basic terminal commands
- Basic arithmetic (+,-,/,*)
- Accepting user input
- For & While loops
- Exception handling
- If-Else statements
- Functions, modules & Imports

Intermediate
- Object oriented programming in Python:Classes, Objects, Methods
- PIP (Pypi)
- List slicing
- String formatting
- Dictionaries & Tuples
- Managing environments
- Dunder methods like __init__

Advanced
- Lambda functions
- Built in libraries like CSV, requests, Sqlite
- Map and Filter
- *args and **kwargs
- Async
- Decorators

Simple Neural Net in Tensorflow.js

Importing tensorflow

```
import * as tf from '@tensorflow/tfjs';
```

Our Neural Net

```
const model = tf.sequential();
model.add(tf.layers.dense({units: 1, inputShape: [1]}));
```

Specifying the loss and the optimizer

```
model.compile({loss: 'meanSquaredError', optimizer: 'sgd'});
```

Synthetic data for training

```
const xs = tf.tensor2d([1, 2, 3, 4], [4, 1]);
const ys = tf.tensor2d([1, 3, 5, 7], [4, 1]);
```

Training the model with the given data

```
model.fit(xs, ys).then(() => {

  model.predict(tf.tensor2d([5], [1, 1])).print();
});
```

Use the model to do inference on a data point the model hasn't seen before

React Redux

@PrasoonPratham

react-redux connect

```
import { connect } from 'react-redux'

YourComponent = connect(
  mapStateToProps,
  mapDispatchToProps
)(YourComponent)

export default YourComponent
```

Store

```
import {
  createStore,
  combineReducers
} from 'redux'
import todos from './todosReducer'
import counter from './counterReducer'

const rootReducer = combineReducers({
  todos,
  counter
})

const store = createStore(rootReducer)

export default store
```

Action Types

```
const ADD_TODO = 'ADD_TODO'
const REMOVE_TODO = 'REMOVE_TODO'
const UPDATE_TODO = 'UPDATE_TODO'
```

action creators

```
const addTodo = (text) => ({
  type: ADD_TODO,
  text
})
const removeTodo = (id) => ({
  type: REMOVE_TODO,
  id
})
const updateTodo = (id, text) => ({
  type: UPDATE_TODO
  id,
  text
})
```

react-redux provider

```
import React from 'react'
import { render } from 'react-dom'
import { Provider } from 'react-redux'
import { createStore } from 'redux'
import todoApp from './reducers'
import App from './components/App'

const store = createStore(todoApp)

render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById('root')
)
```

reducers

```
const initialState = {
  todos: []
}

function todosReducer(state = initialState, action) {
  switch (action.type) {
    case UPDATE_TODO:
      const newState = deepClone(state)
      const todo = newState.todos.find(
        todo => todo.id === action.id
      )
      todo.text = action.text
      return newState
  }
}

function deepClone(obj) {
  return JSON.parse(JSON.stringify(obj))
}
```

Find Null values

```python
import pandas as pd

dataframe = pd.read_csv("data.csv")

for col in dataframe.columns:
    print(col , ":" , train_data[col].isnull().sum())

#Output
# Pclass : 0
# Sex : 0
# Age : 177
# SibSp : 0
# Parch : 0
# Fare : 0
# Embarked : 2
```

Fill NaN values with median

```
import pandas as pd

dataframe = pd.read_csv("data.csv")

dataframe['Age'] = dataframe['Age'].fillna(dataframe['Age'].median())

#Output
# Fills NaN values in the age column with the median of that column
```

Drop columns from dataframe

```
import pandas as pd

dataframe = pd.read_csv("data.csv")

dataframe.drop(['PassengerId','Age' ], axis = 1, inplace = True)

#Output
# Drops columns from your dataframe
```

Change Type of array

```python
import numpy as np

list = np.array([1.2,3.5])

list = list.astype(int)

#Output
# list = [1,3]
```

Train-Test Split

```
from sklearn.model_selection import train_test_split

trainData , valData, trainLabels, valLabels = (train_test_split(train_data, train_labels,train_size=0.76))

# Output
# Splits data into training and validation datasets
```

Make submissions file

```
import pandas as pd

submission = pd.read_csv('gender_submission.csv')
submission['Survived'] = pred
submission.to_csv('submission.csv', index=False)

# Output
# Submission.csv is created with predictions from our model
```