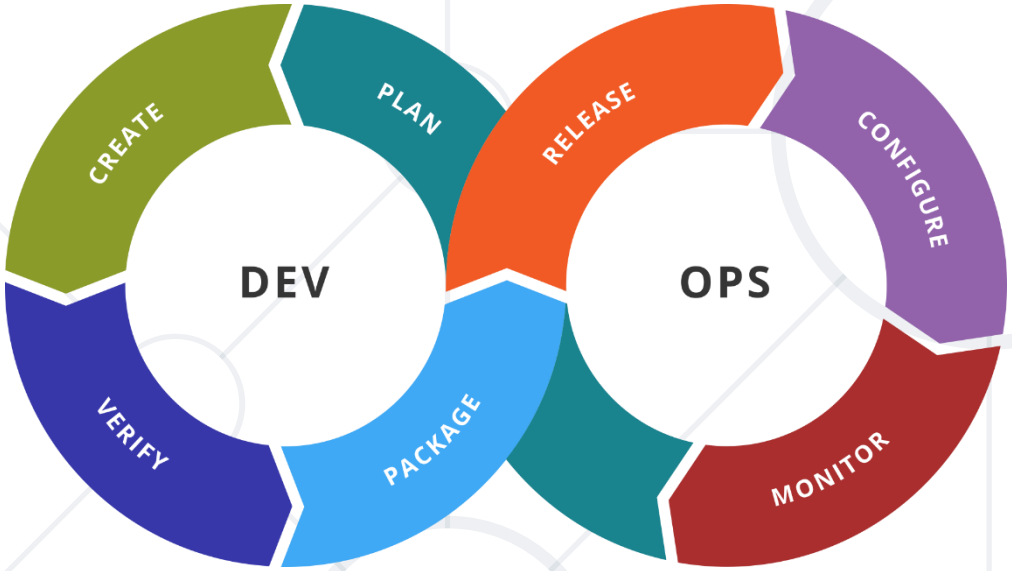
Salt

Introduction and Basic Techniques



SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.org

You Have Questions?

Software University

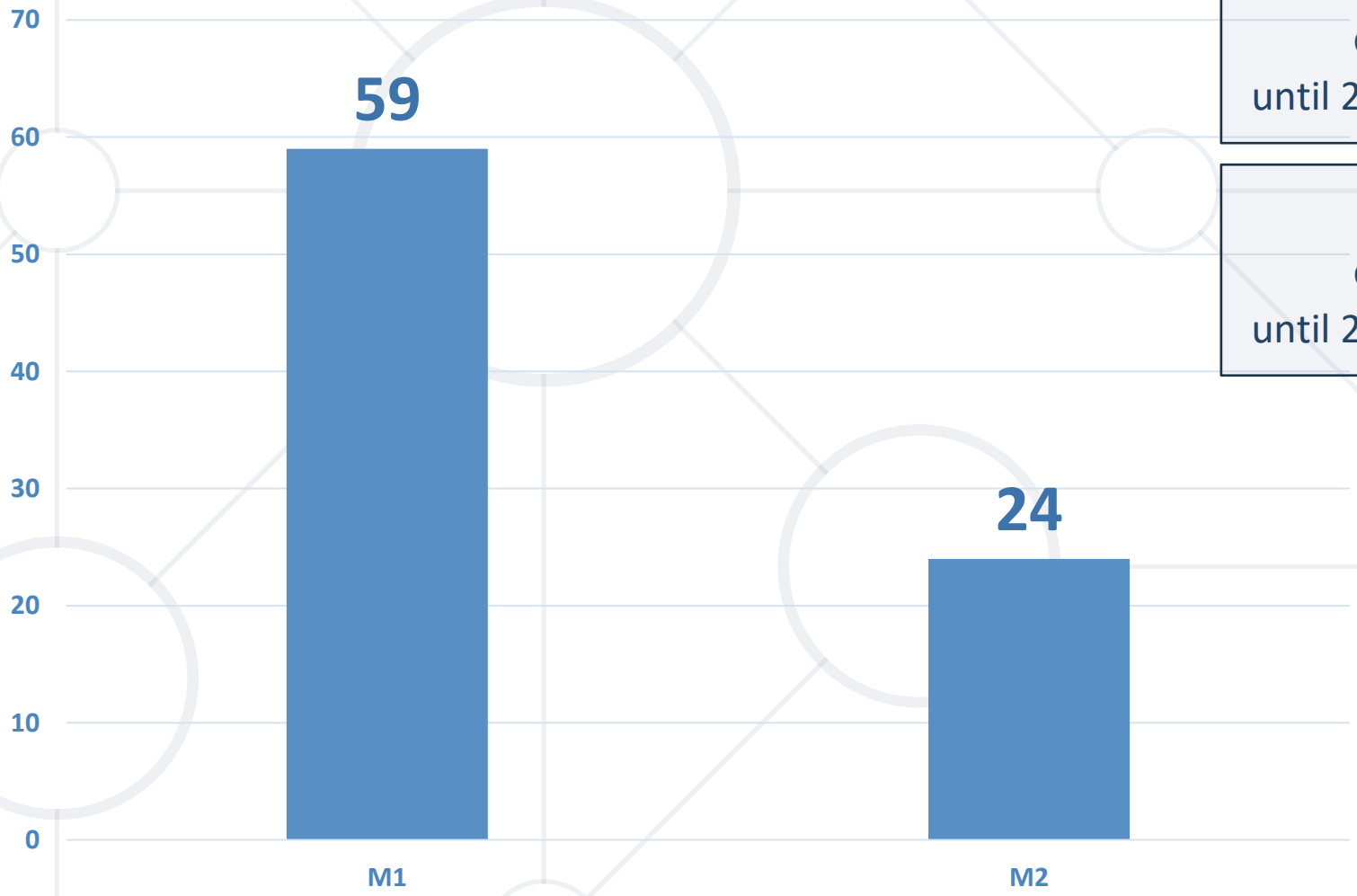
sli.do

#DevOps-23

facebook.com/groups

/DevOpsInfrastructureandConfigManagementApril2023

Homework Progress



Solutions for M2
can be submitted
until 23:59 on 19.04.2023

Solutions for M3
can be submitted
until 23:59 on 26.04.2023

Previous Module (M2)

Quick Overview

What We Covered

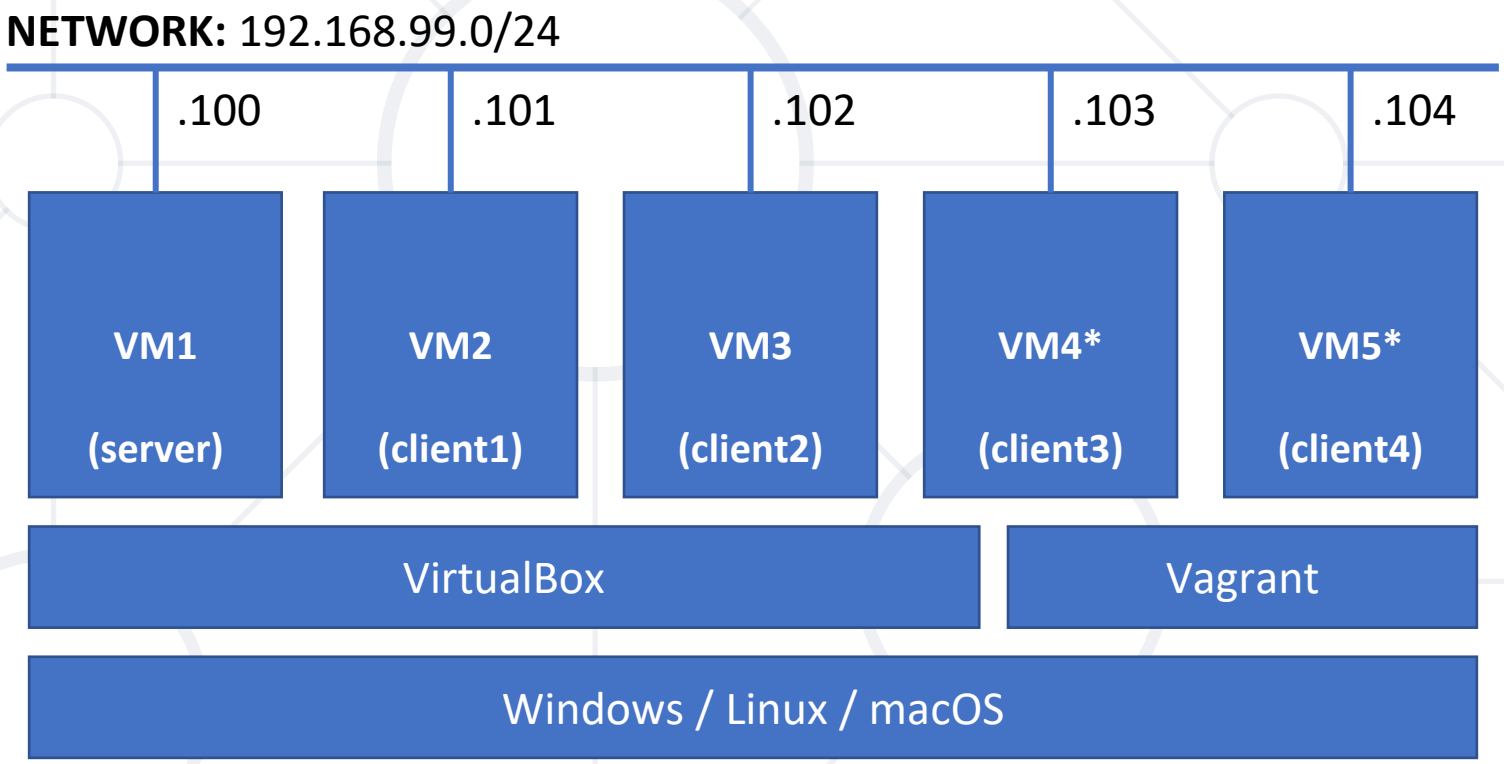
1. Introduction to Ansible
   - Other solutions
   - Ansible architecture
2. Working with Ansible
   - Work with Inventories and Configurations
   - Using Modules
3. Advanced Ansible
   - Playbooks and Roles

This Module (M3)

Topics and Lab Infrastructure

Table of Contents

Lab Infrastructure



* VM4 and VM5 can be skipped. Of course, the exercises should be adjusted accordingly

Salt 101

Introduction. Architecture. Installation

Introduction

- Infrastructure management solution that can be used for **data-driven orchestration**, **remote execution**, and **configuration management**
- Management instructions are written in **YAML**
- Two operation modes – with **agents** (**minions**) and agent-less
- Linux/Unix/Windows are both supported as **minions** and **masters**
- Installable via **package management system** and **bootstrap script**

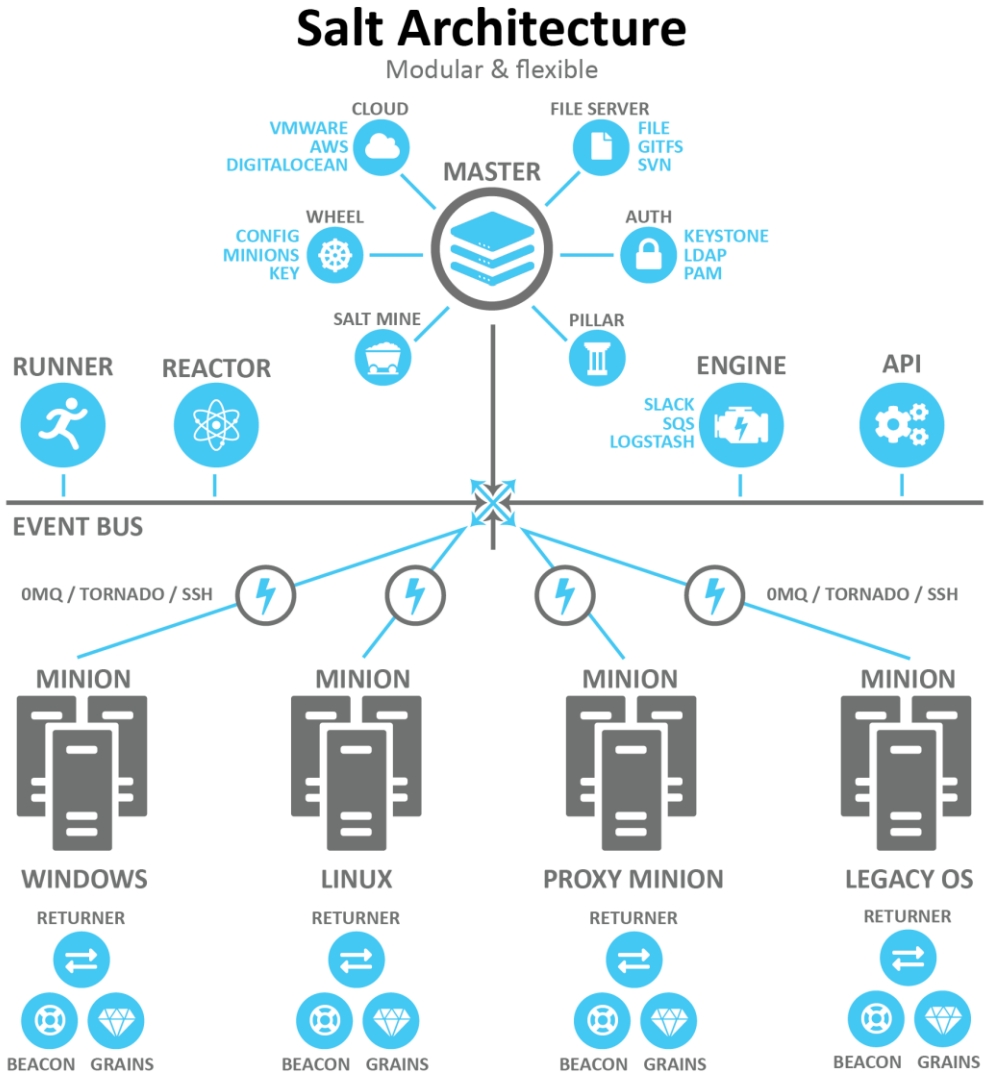
Components and Workflow

- **Salt master** is the machine that controls the infrastructure and dictates policies for the servers it manages
- Master operates both as a **repository** for **configuration data** and as the **control center** that initiates **remote commands** and ensures the **state** of your other machines
- Servers that Salt manages are called **minions**
- Minions are responsible for **executing the instructions** sent by the master, **reporting** on the success of jobs, and **providing data** about the underlying host

Artifacts

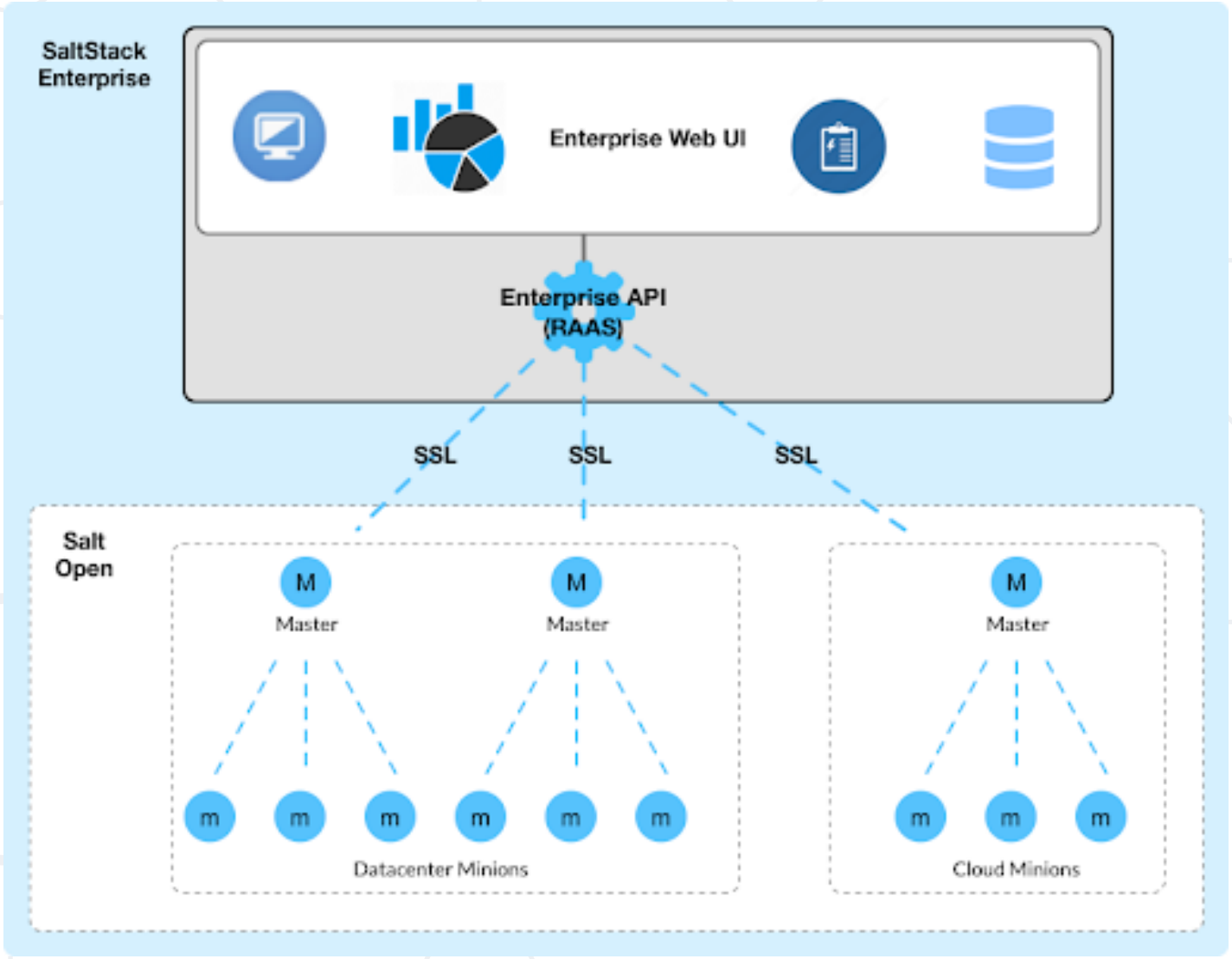
- **Execution modules** are sets of related **functions** that perform work on minions
- The configuration management portion of Salt is primarily implemented using the **state** system, which uses **state modules**
- Salt **formulas** are **sets of state module calls**, arranged with the aim of producing a certain result
- Configuration management is handled by **SLS files**, written in **YAML**, that describe how a system should look once the formula has been applied
- Salt **grains** are pieces of information, gathered by and maintained by a minion, primarily concerning its underlying host system

General Salt Architecture



https://docs.saltproject.io/salt/user-guide/en/latest/topics/overview.html

SaltStack Config



https://docs.saltproject.io/salt/user-guide/en/latest/topics/overview.html

Practice: Salt 101

Live Demonstration in Class

Salt 102

Files. Templates. Filtering. Pillars and Beacons

General Files Outline

- Main Salt components configuration files
  - **/etc/salt/master** and **/etc/salt/minion**
- State files describe the desired state for the minions
  - **YAML** based **SLS** files stored in **/srv/salt** by default
- Top file (**top.sls**) maps minions to states
- Configuration can be split amongst several state files

Filtering (or targeting)

- Can be applied when the configuration is deployed
- Done either in the top file or during the execution
- May use the full name or part of the name plus globing
- Regular expressions are also available
- List of minions can also be used as a target
- IP addresses or subnet addresses can be used as well
- Grain information can be used as well

Pillars

- They are similar to the grains but are created on the master
- Contain information about a minion or group of minions
- Can be used to store and send sensitive data
- Or to hold a variable data
- Usually stored in /srv/pillar folder
- Follow the same structure as with the standard config files
- Use the SLS extension and have a top file as well

Beacons

- Use the Salt event system to monitor non-Salt processes
- Minions monitor and report any activity
- Support monitoring of activities like
  - **File system changes** and **service status**
  - **Network** and **disk usage**, etc.
- Configured either in **/etc/salt/minion**
- Or in **/etc/salt/minion.d/beacons.conf**
- Require the **Pyinotify** package to be present on the minions

Reactors

- Ability to trigger actions in response to an event
- Bind SLS files to event tags on the master
- Have two parts
  - Reactor option set in the master configuration file
  - Reaction files use highdata (like the state system) to define reactions to be executed
- Usually stored in **/srv/reactor/**
- Can be **local**, **runner**, **wheel**, or **caller**

Practice: Salt 102

Live Demonstration in Class

Salt 103

Custom Modules

General #1

- Regular **Python** modules
- Stored in a directory called **_modules/** within the **file_roots** specified by the master config file
- Synced to the minions when **state.highstate** is run
- Or by executing the **saltutil.sync_modules** or **saltutil.sync_all** functions
- If having the same name with a system module, they will take the place of the default module with the same name

General #2

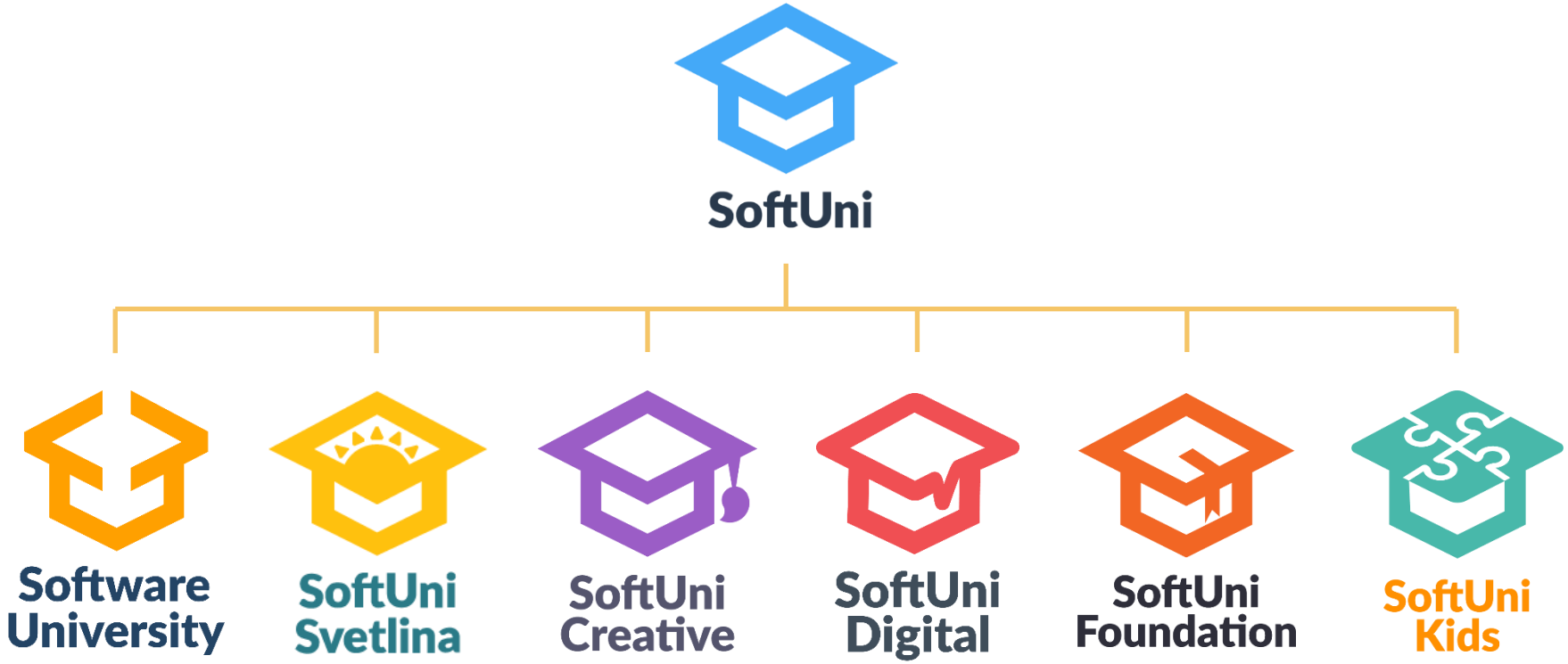
Software University

- Module's default name is its **filename**
- The **`__virtualname__`** variable definition defines a custom name for the module
- Its is returned by the **`__virtual__`** function
- Documentation can be added by adding **Python docstring** to the function
- We can have private functions (start with _) which are not visible outside the module. For example, **`_calc()`**
- Salt sees the public functions. For example, **`get()`**. They are available as **`module.get()`**

Practice: Salt 103

Live Demonstration in Class

Questions?



SoftUni Diamond Partners

Software University

SCHWARZ

LIDL

Kaufland

SCHWARZ Produktion

prezero

Coca-Cola HBC Bulgaria

POKERSTARS | CAREERS

Postbank

Решения за твоето утре

AMBITIONED

DXC TECHNOLOGY

SOFTWARE GROUP

Bosch.IO

INDEAVR

Serving the high achievers

DRAFT KINGS

PHAR VISION

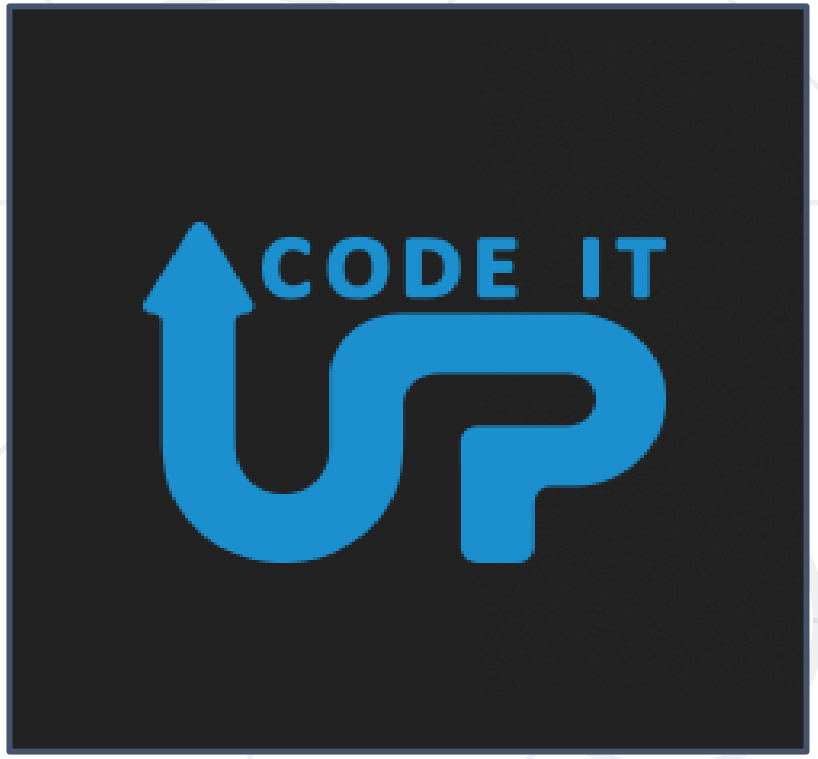
SmartIT

createX

SUPER HOSTING .BG

Educational Partners

Software University



License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://softuni.org
- © Software University – https://softuni.bg

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Pr ofession and Job for Software Developers
  - softuni.bg, softuni.org
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg