

SOFTWARE ENGINEERING

---

**DEVELOPMENT**

# FACETS OF SOFTWARE ENGINEERING

- ▶ Requirements
- ▶ Design & Architecture
- ▶ **Implementation**
- ▶ Quality Assurance
- ▶ Documentation
- ▶ Packaging & Delivery
- ▶ Maintenance & Support



Team Process

---

**LIBRARIES**

# LIBRARIES

- ▶ 3rd party code used by an application
- ▶ Come from:
  - ▶ Paid License
  - ▶ Open Source
  - ▶ Internally Developed
- ▶ Used at:
  - ▶ Build
  - ▶ Runtime
  - ▶ Testing

# LIBRARIES

- ▶ Often referred to as a “dependency”
- ▶ Accessed through its Application Programming Interface (API)
  - ▶ Set of methods/functions exposed by a library that are used to interact with it
  - ▶ Versioned (hopefully)
  - ▶ Documented (again, hopefully)

## LIBRARIES (HEAT PROJECT)

```
# The order of packages is significant, because pip processes them in the order
# of appearance. Changing the order has an impact on the overall integration
# process, which may cause wedges in the gate later.

pbr>=1.6 # Apache-2.0
Babel>=1.3 # BSD
croniter>=0.3.4 # MIT License
cryptography>=1.0 # BSD/Apache-2.0
debtcollectors>=1.2.0 # Apache-2.0
eventlet!<=0.18.3, >=0.18.2 # MIT
greenlet>=0.3.2 # MIT
KeystoneMiddleware!<=4.1.0, >=4.0.0 # Apache-2.0
Twisted>=2.3 # BSD
netaddr!<=0.7.16, >=0.7.12 # BSD
oslo.cache>=1.5.0 # Apache-2.0
oslo.config>=3.7.0 # Apache-2.0
oslo.concurrency>=3.5.0 # Apache-2.0
oslo.context>=0.2.0 # Apache-2.0
oslo.db>=4.1.0 # Apache-2.0
oslo.i18n>=2.1.0 # Apache-2.0
oslo.log>=1.14.0 # Apache-2.0
oslo.messaging>=4.0.0 # Apache-2.0
oslo.middleware>=3.0.0 # Apache-2.0
oslo.policy>=0.5.0 # Apache-2.0
oslo.reports>=0.6.0 # Apache-2.0
oslo.serialization>=1.10.0 # Apache-2.0
oslo.service>=1.0.0 # Apache-2.0
oslo.utils>=3.5.0 # Apache-2.0
osprofiler>=1.1.0 # Apache-2.0
oslo.versionedobjects>=1.5.0 # Apache-2.0
PasteDeploy>=1.5.0 # MIT
pycrypte>=2.6 # Public Domain
python-barbicanclients>=3.3.0 # Apache-2.0
python-cellaclient>=2.2.1 # Apache-2.0
python-cinderclient>=1.3.1 # Apache-2.0
python-designateclient>=1.5.0 # Apache-2.0
python-glanceclient>=2.0.0 # Apache-2.0
python-heatclient>=0.6.0 # Apache-2.0
python-keystoneclient!<=1.8.0, !=2.1.0, >=1.5.0 # Apache-2.0
python-magnumclient>=0.2.1 # Apache-2.0
python-manilaclient>=1.3.0 # Apache-2.0
python-mistralclient>=1.0.0 # Apache-2.0
python-neutronclient!<=4.1.0, >=2.6.0 # Apache-2.0
python-novaclient!<=2.33.0, >=2.29.0 # Apache-2.0
python-openstackclient>=2.1.0 # Apache-2.0
python-saharacclient>=0.13.0 # Apache-2.0
python-senlinclient>=0.3.0 # Apache-2.0
python-swiftclient>=2.2.0 # Apache-2.0
python-troveclient>=1.2.0 # Apache-2.0
python-zaqabclient>=0.3.0 # Apache-2.0
pytz>=2013.6 # MIT
"requirements.txt" 61L, 2204C
```

# LIBRARY ARCHETYPES

- ▶ Web Frameworks
  - ▶ Drives the flow of a web application from page display and user data down to the storage layer
- ▶ Object-Relational Mapping Frameworks
  - ▶ Handles the conversion between the in-memory object representation of data and the persistent storage (e.g. database)
- ▶ Testing Frameworks
  - ▶ Runs test code, ensuring isolation and providing reports on pass/fail and coverage

# LIBRARY ARCHETYPES

- ▶ Web Components
  - ▶ Widgets to facilitate interactive or asynchronous web interfaces
  - ▶ Menus, search bar population, etc.
- ▶ Parsers / Serializers
  - ▶ Conversion between in-memory object representations of data and a specific, standardized format
  - ▶ JSON, YAML, XML, etc.



# LIBRARY ARCHETYPES

- ▶ User & Role Management
  - ▶ Authorization and authentication for users
- ▶ Service Client Libraries
  - ▶ Used for connecting to some service external from your application
  - ▶ Facebook, Twitter, etc.
- ▶ Utilities
  - ▶ Any number of other reusable functions

---

**LOGGING**

# LOGGING

- ▶ An application's way of producing meaningful information for end users, developers, and support engineers
- ▶ Used for a wide range of topics, including:
  - ▶ current state (running operations, scale, etc.)
  - ▶ health (load, bottlenecks, etc.)
  - ▶ debugging (variable values, etc.)

# LOGGING

- ▶ Severity can be tuned to display different granularity of messages
  - ▶ debug v. info v. fatal
- ▶ Output can be tuned to different locations
  - ▶ console, log file, socket
- ▶ Log files should have rotation policies to control size and scope
  - ▶ size, timestamp, etc.

# LOGGING CONCERNS

- ▶ Verbosity
  - ▶ Data within the logs should be concise enough to be understood
- ▶ Security
  - ▶ Do not display sensitive data (i.e. **no** passwords) or commands

## LOGGING CONCERNS

- ▶ History
  - ▶ How long until log files are automatically reaped/rotated?
- ▶ Size
  - ▶ How much space will the logs potentially occupy?

# LOGGING CONCEPTS

- ▶ Handler
  - ▶ Decides what to do with the logged message
  - ▶ file, console, socket
- ▶ Formatter
  - ▶ Extra data to include with the log message
  - ▶ timestamp, severity, file, line number, stacktrace/traceback
- ▶ Severity
  - ▶ Metadata indicating the important of the log message
  - ▶ DEBUG, INFO, WARN, ERROR, FATAL/CRITICAL

TEXT

---

# LOGGING EXAMPLE



---

# CONCURRENCY

# CONCURRENCY

- ▶ Way to support multiple, concurrent users in an application
- ▶ Implemented as multiple “threads” executing in a single “process”

# CONCURRENCY

- ▶ A thread is a context for a particular user
- ▶ Threads execute “concurrently” (but not really)
- ▶ Prevents blocking an entire application on a single, long running operation
- ▶ Almost always non-deterministic
  - ▶ Multiple runs may behave in drastically different ways

# CONCURRENCY CONCERNS

- ▶ Deadlocking
  - ▶ When a thread is waiting on a resource that never becomes available
- ▶ Scaling
  - ▶ Too many threads can increase the size & resources of the application

## CONCURRENCY CONCERNS

- ▶ Shared State
  - ▶ Concurrent access to the same data
  - ▶ database, shared memory, etc.
- ▶ Race Condition
  - ▶ Non-deterministic results depending on the order in which threads execute

TEXT

---

# CONCURRENCY EXAMPLE

---

**INTERNATIONALIZATION**

# INTERNATIONALIZATION

- ▶ Internationalization (i18n)
  - ▶ The process of making an application **capable** of supporting multiple locale
  - ▶ Implemented as hooks to translate user-readable data between languages
- ▶ Localization (l10n)
  - ▶ Process of adding support/configuration for a particular locale
  - ▶ Involves language translation, units, currency, etc.



TEXT

---

# INTERNATIONALIZATION EXAMPLE

---

# JAVADOC EXAMPLE

## HOMework

- ▶ Quiz
- ▶ Project: Documentation