



BY PETER ULLRICH

DESIGN PATTERNS IN PYTHON

RAW CODE

The screenshot shows a code editor with two files open:

- account.py**: A Python file containing logic for account creation and verification. It imports logging, flask, and various utility modules. The class `AccountLogic` contains methods for registering users and validating their accounts via email.
- callback.py**: A Python file defining a `Callback` class that implements a `__init__(self, target, category)` method. This method sets `_METHOD_URL` to "URL", `method` to "POST", `target` to "https://www.google.com/api/1/page/callbacks", and `category` to "POST". It also defines a `Callback` class with its own `__init__(self, target, category)` method, which sets `_METHOD_URL` to "URL", `self.target` to `target`, and `self.category` to `category`.

```
account.py (left pane):
```

```
import logging
from flask import jsonify, session
from flask.decorators import deco
import bcrypt
from flask_bcrypt import generate_password_hash as bhash
from app import const, errors
from http.cookies import SimpleCookie
from model import EncryptedData, User
from util import security

logger = logging.getLogger(__name__)

class AccountLogic:
    @classmethod
    def decode_json(cls, dm, chat_id, pw, api_key):
        logger.info("Received new register request")

        user_exists = AccountLogic._check_user_exists(dm.chat_id)
        if user_exists:
            return errors.USER_EXISTS, None

        api_env = dm.get_api_env(dm.chat_id)
        api_conf = dm.get_api_conf(api_env, api_key)

        user_salt_pw = security.decrypt_user_key(pw, user_exists)
        pw_encrypt = dm.get_password(dm.chat_id)

        api_env_encrypted = security.encrypt(api_env, salt=user_salt_pw)
        auth_data = EncryptedData(user_auth=user_salt_pw, pw=pw_encrypt)
        api_data = encrypteddata(api_env_encrypted, salt=user_salt_pw)

        user_new = User(dm.chat_id, auth_data, api_data)

        try:
            user.add(user_new)
            return "ACCOUNT CREATED", 201
        except Exception as e:
            return str(e), 500

    @classmethod
    def login(chat_id, pw):
        user = User.query.filter_by(chat_id=chat_id).first()

        if user is None:
            return errors.PW_WRONG_OR_NO_ACCOUNT, 401

        pw4 = security.decrypt_user_key(user.auth.salt, user.auth.pw)

        if pw4 != user.auth.value:
            return errors.PW_WRONG_OR_NO_ACCOUNT, 401

        token = SECURITY.gen_token()
        api_conf = security.decrypt(user.api_conf.value, salt=1, user.api_conf.salt)

        session.jwt.set(AUTH_TOKEN) = token
        session.conf(API_CONF) = api_conf

        return jsonify({const.AUTH_TOKEN: token}), 200

    @classmethod
    def decode_json(cls, dm, chat_id, pw):
        user_salt_pw = dm.get_password(dm.chat_id)
        print(user_salt_pw)

        user_info = dm.get_user_list(chat_id)
        print(user_info)
```

```
callback.py (right pane):
```

```
from sqlalchemy import Column, ForeignKey, Integer, String
from model import Base, BaseModel
_METHOD_URL = "URL"

class Callback(Base, BaseModel):
    account_id = Column(Integer, ForeignKey("account.reference_id"))

    method = Column(String, nullable=False)
    target = Column(String, nullable=False)
    category = Column(String, nullable=False)

    def __init__(self, target, category):
        self.method = _METHOD_URL
        self.target = target
        self.category = category

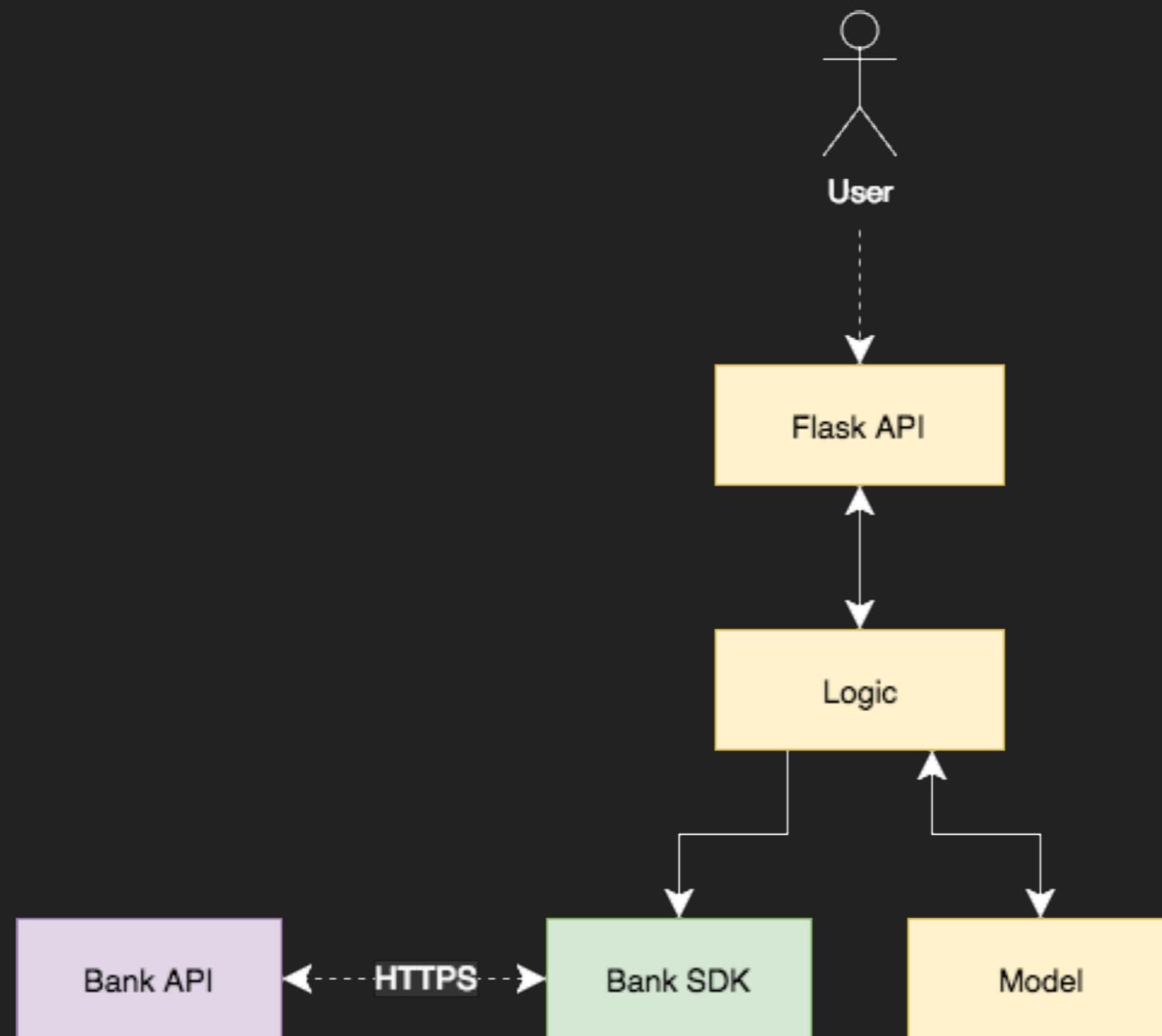
    @property
    def target(self):
        return self.target

    @target.setter
    def target(self, target):
        self.target = target

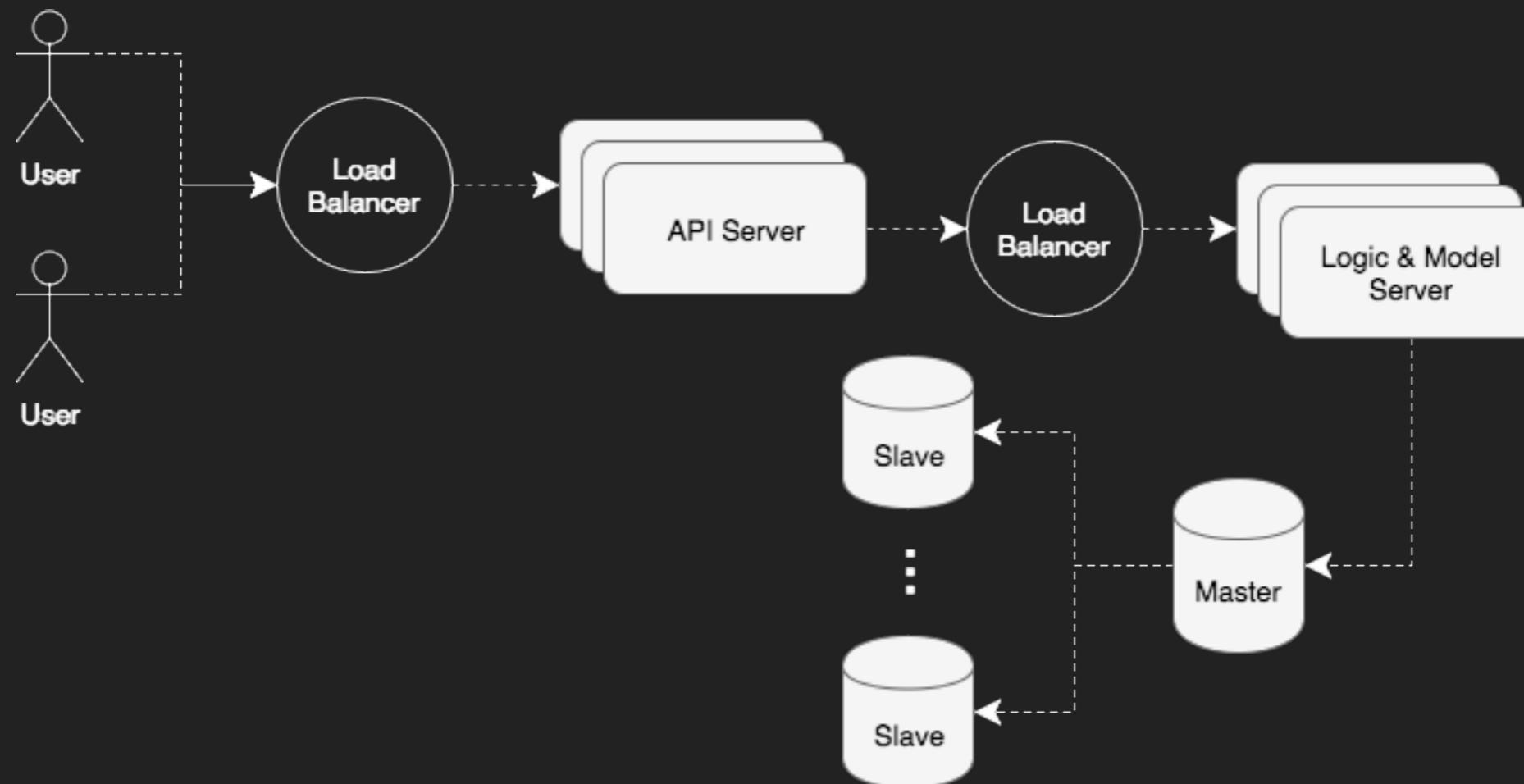
    @property
    def category(self):
        return self.category

    @category.setter
    def category(self, category):
        self.category = category
```

SOFTWARE DESIGN



SOFTWARE ARCHITECTURE



A SHORT HISTORY OF SOFTWARE DESIGN



Christopher Alexander (Real-world Architect)

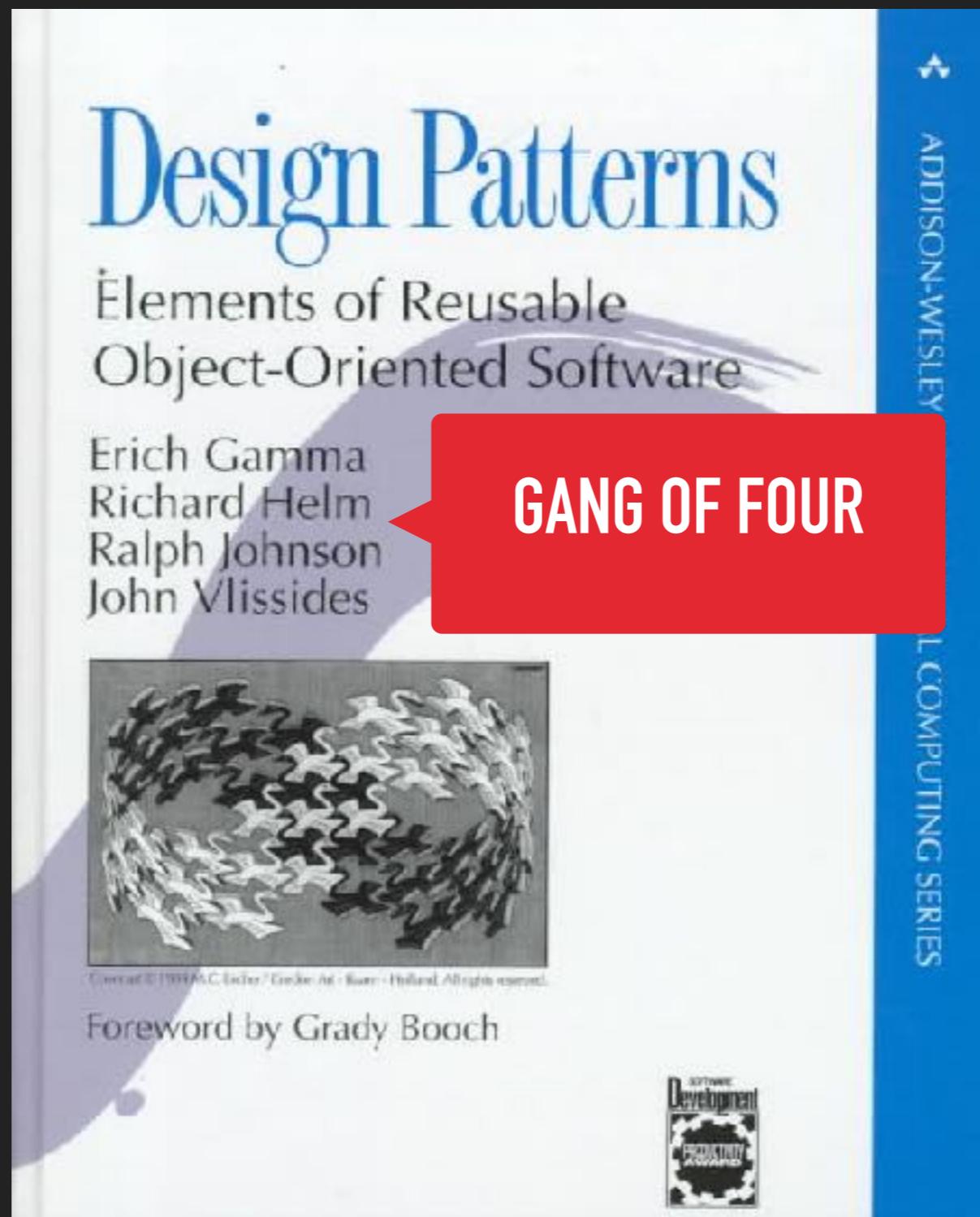
253 - 1977

A Pattern Language

Towns • Buildings • Construction



Christopher Alexander
Sara Ishikawa • Murray Silverstein
WITH
Max Jacobson • Ingrid Fiksdahl-King
Shlomo Angel



1994

Image attribution: [wikipedia.org](https://en.wikipedia.org)

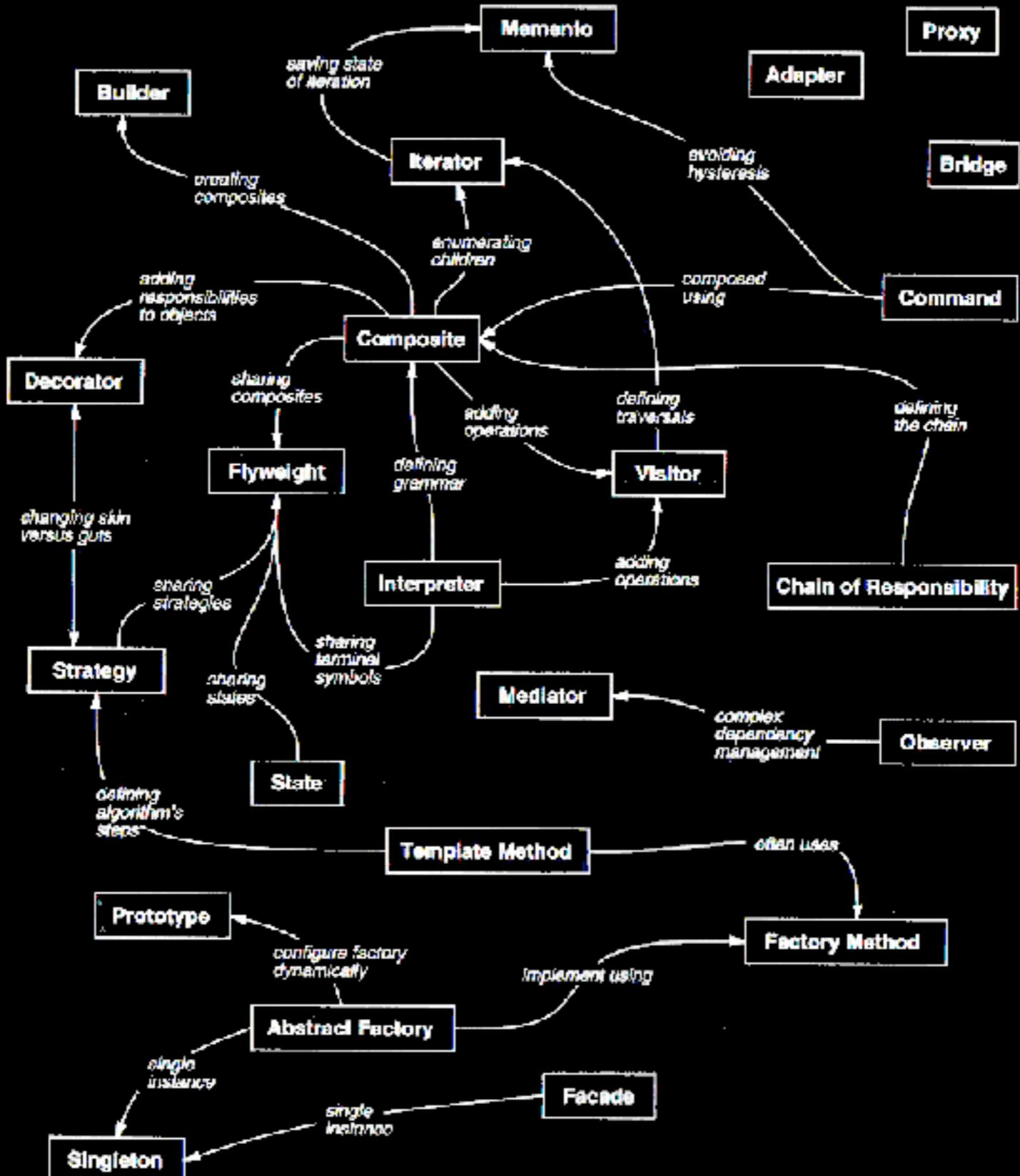


Figure 1.1: Design pattern relationships

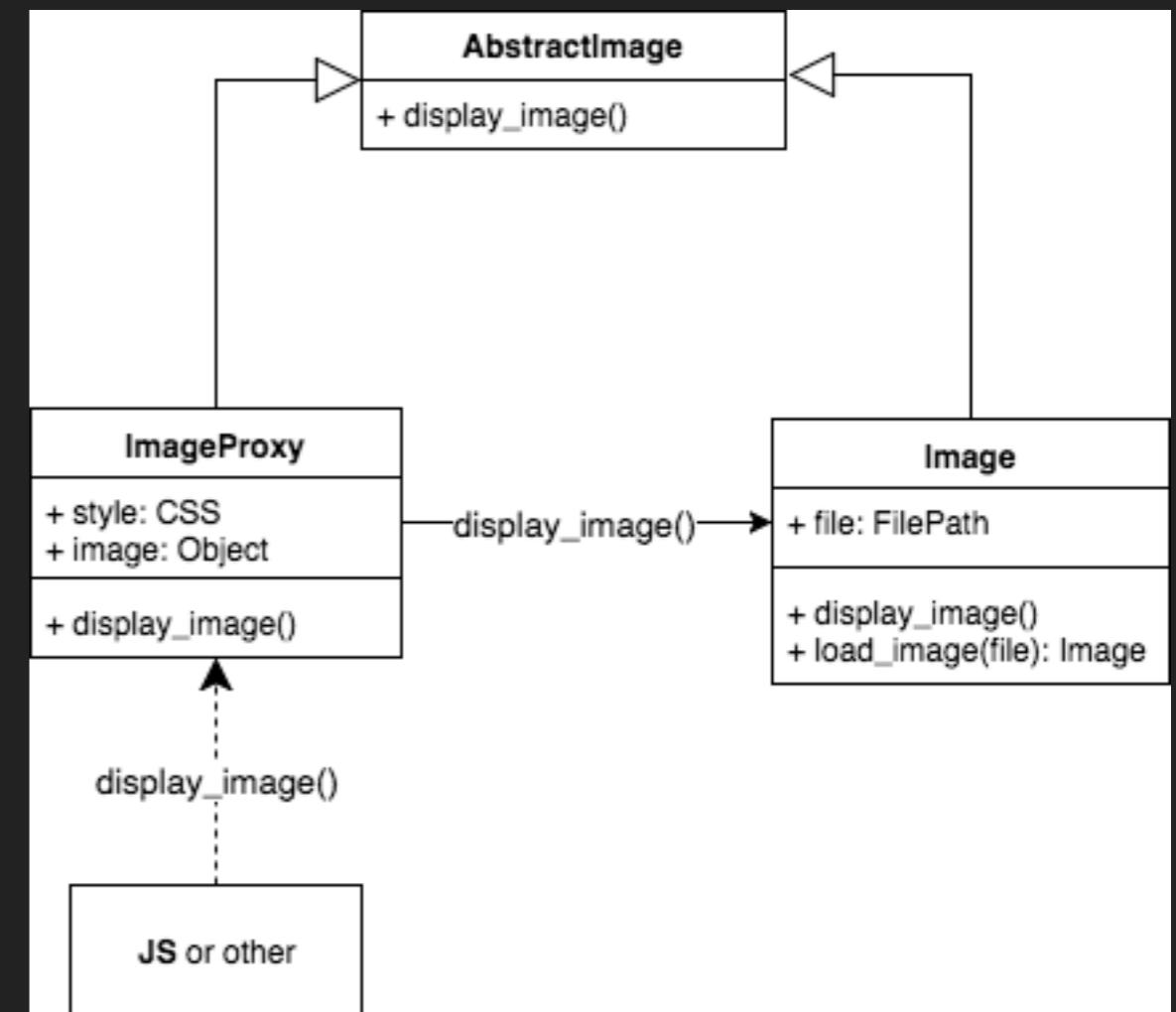
**WHAT IS A
DESIGN PATTERN**

PROXY PATTERN

- ▶ Problem
 - ▶ high-res images on website
 - ▶ Long loading time
 - ▶ Style images
- ▶ Solution
 - ▶ Replace with placeholders (proxies)
 - ▶ Style placeholders instead

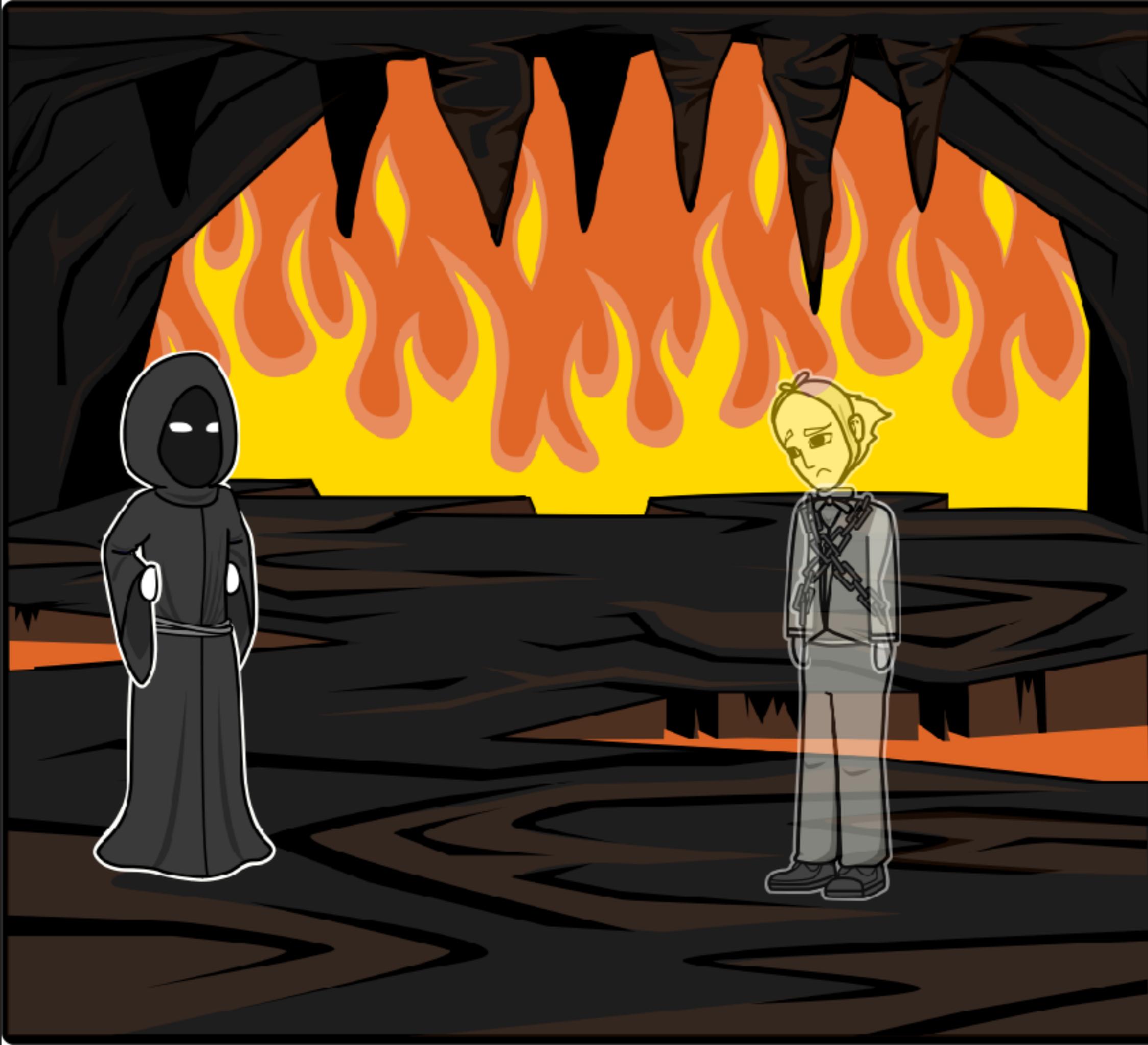
PROXY PATTERN

- ▶ Problem
 - ▶ high-res images on website
 - ▶ Long loading time
 - ▶ Style images
- ▶ Solution
 - ▶ Replace with placeholders (proxies)
 - ▶ Style placeholders instead



HOW TO USE DESIGN PATTERNS

A SHORT STORY





Create 3 updates which give me
sudo-rights over iPhone user



As you wish, boss

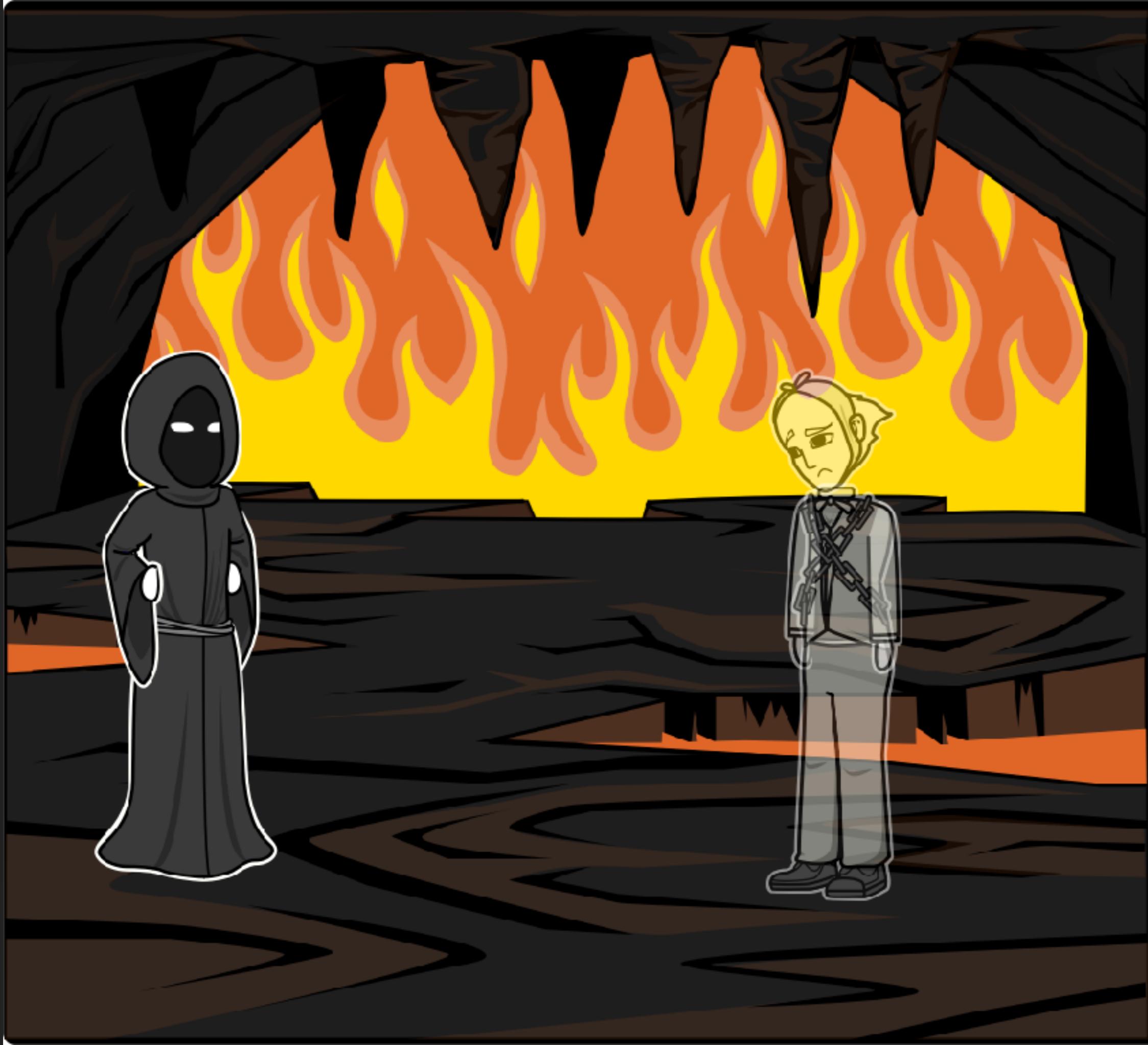


iPhone User



Android User

**We also want
that update!**





I need 7 more updates to control the Android user!





A cartoon illustration set in a dark, rocky landscape. In the center, a character with a pale face and short hair stands with arms crossed, looking towards the left. To the left, another character wearing a dark hooded cloak stands with hands on hips. Behind them is a large, bright orange fire with yellow flames, casting a glow on the surrounding rocks. A speech bubble originates from the central character, containing the text "Today was refactor-day, but I'll get right to it."

**Today was refactor-day,
but I'll get right to it.**





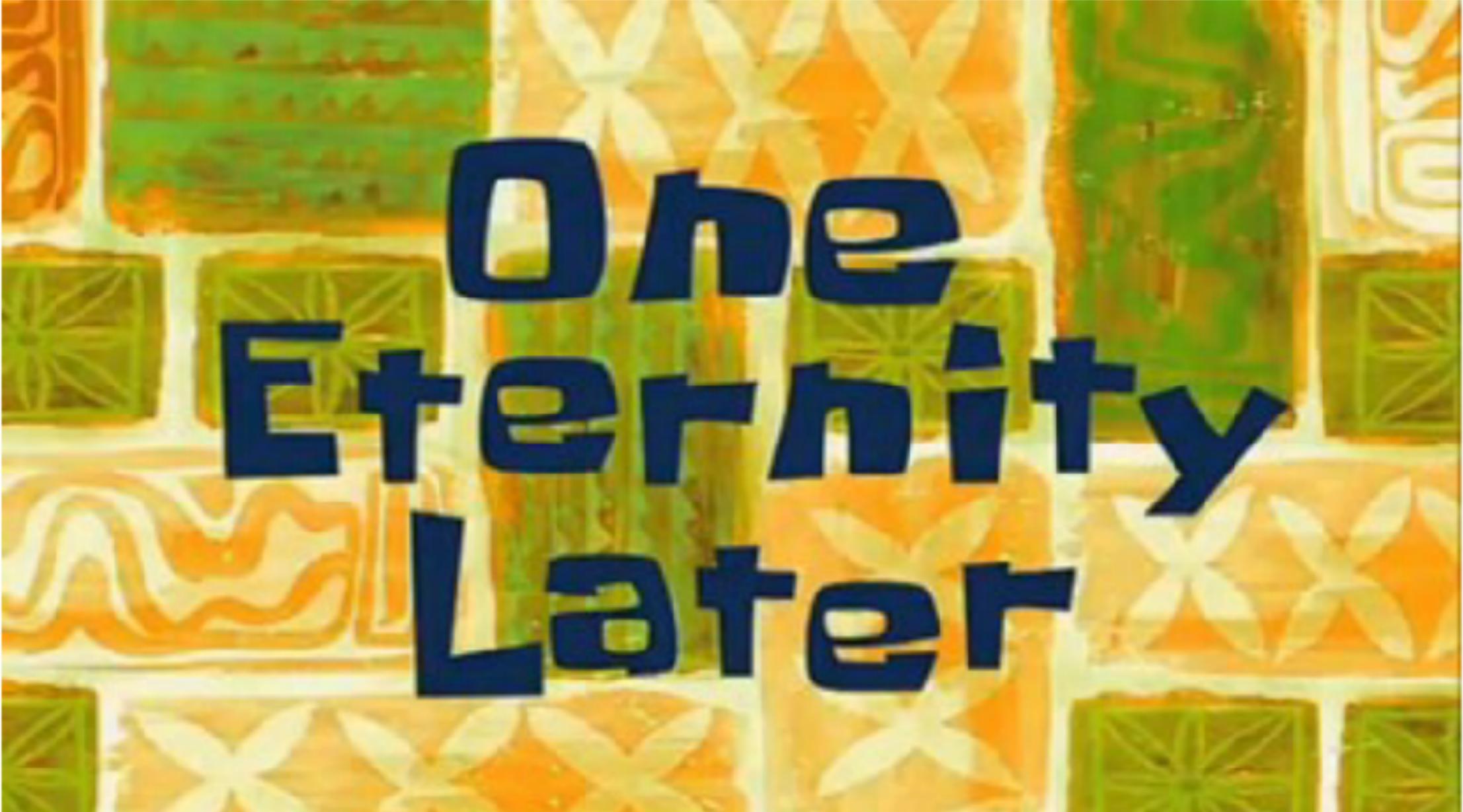
Windows Phone User



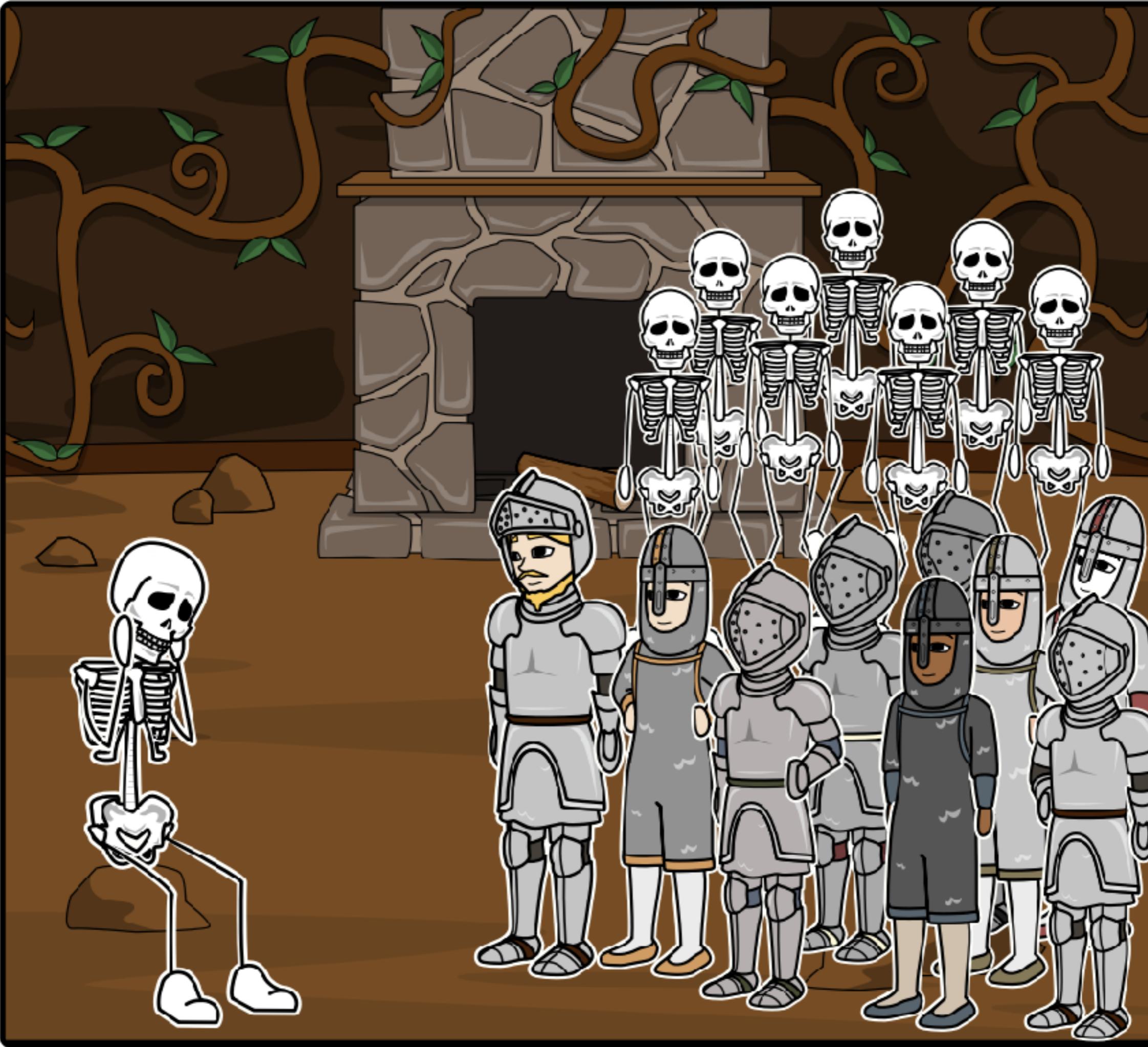
Ehm...

Would you mind developing this update for Windows Phone as well?





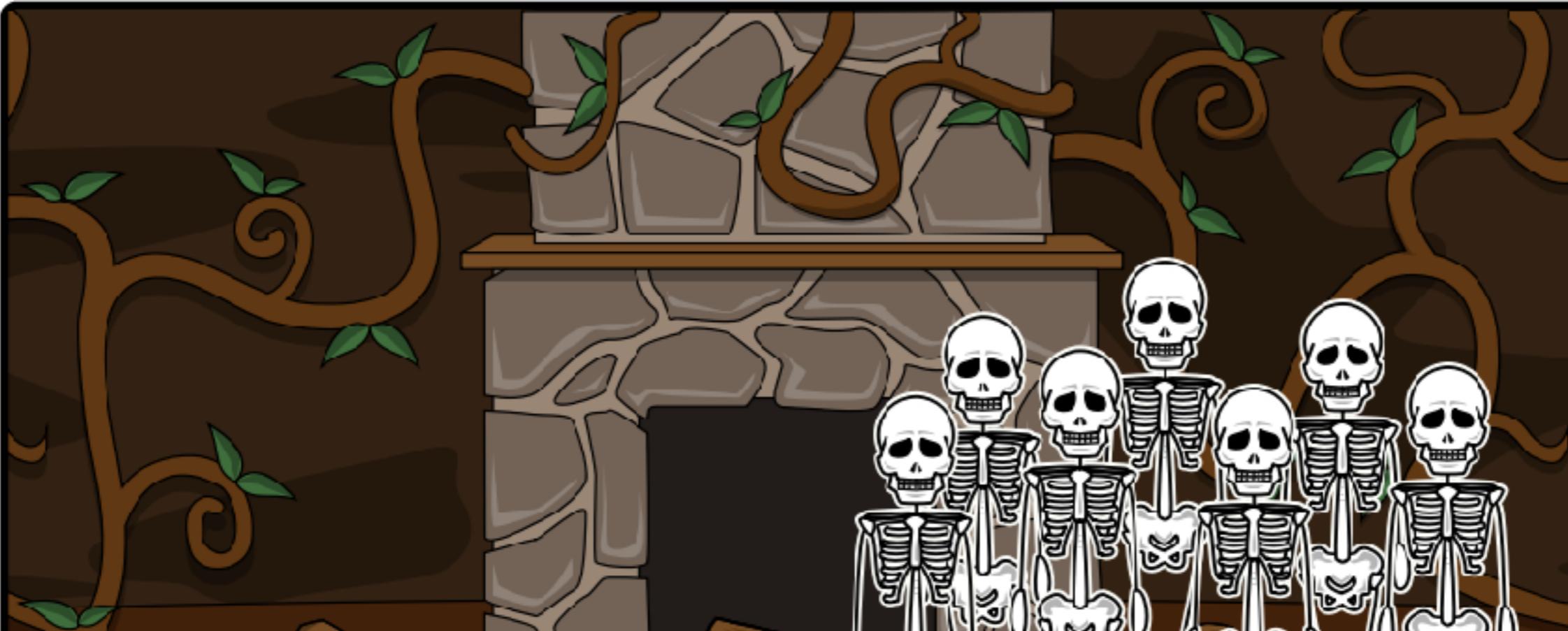
**One
Eternity
Later**





A cartoon illustration set in a dark, subterranean environment. In the foreground, a single skeleton stands on the left, facing right. Behind it, a group of approximately ten skeletons in various states of decay are gathered. Some are standing upright, while others are leaning against walls or each other. The background features stone walls with vines growing over them, and a shelf with some items. A speech bubble in the upper center contains the text "Thank you".

Thank you

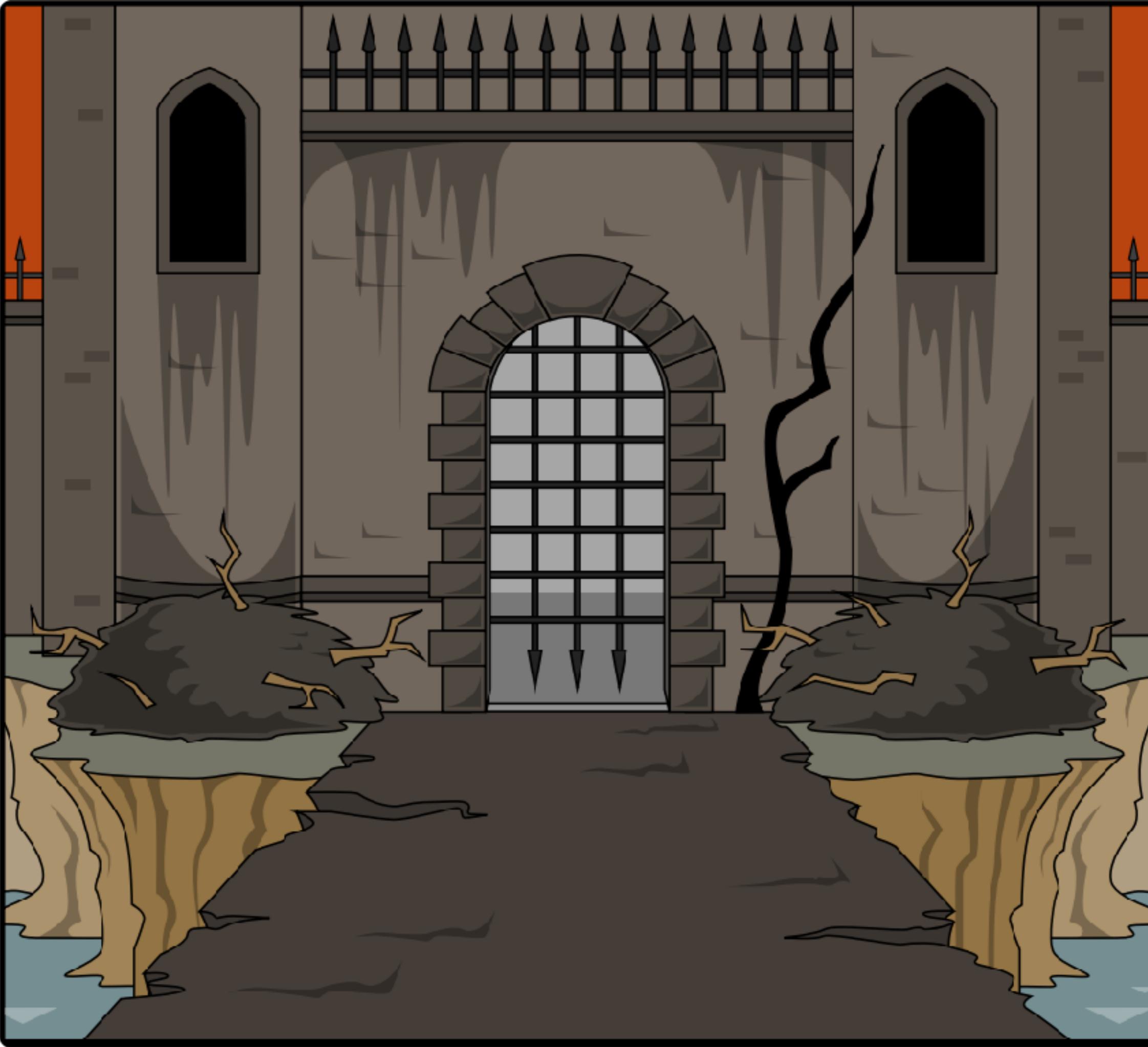


Windows

A fatal exception OE has occurred at 0028:C562F1B7 in UXD ctpci9x(05)

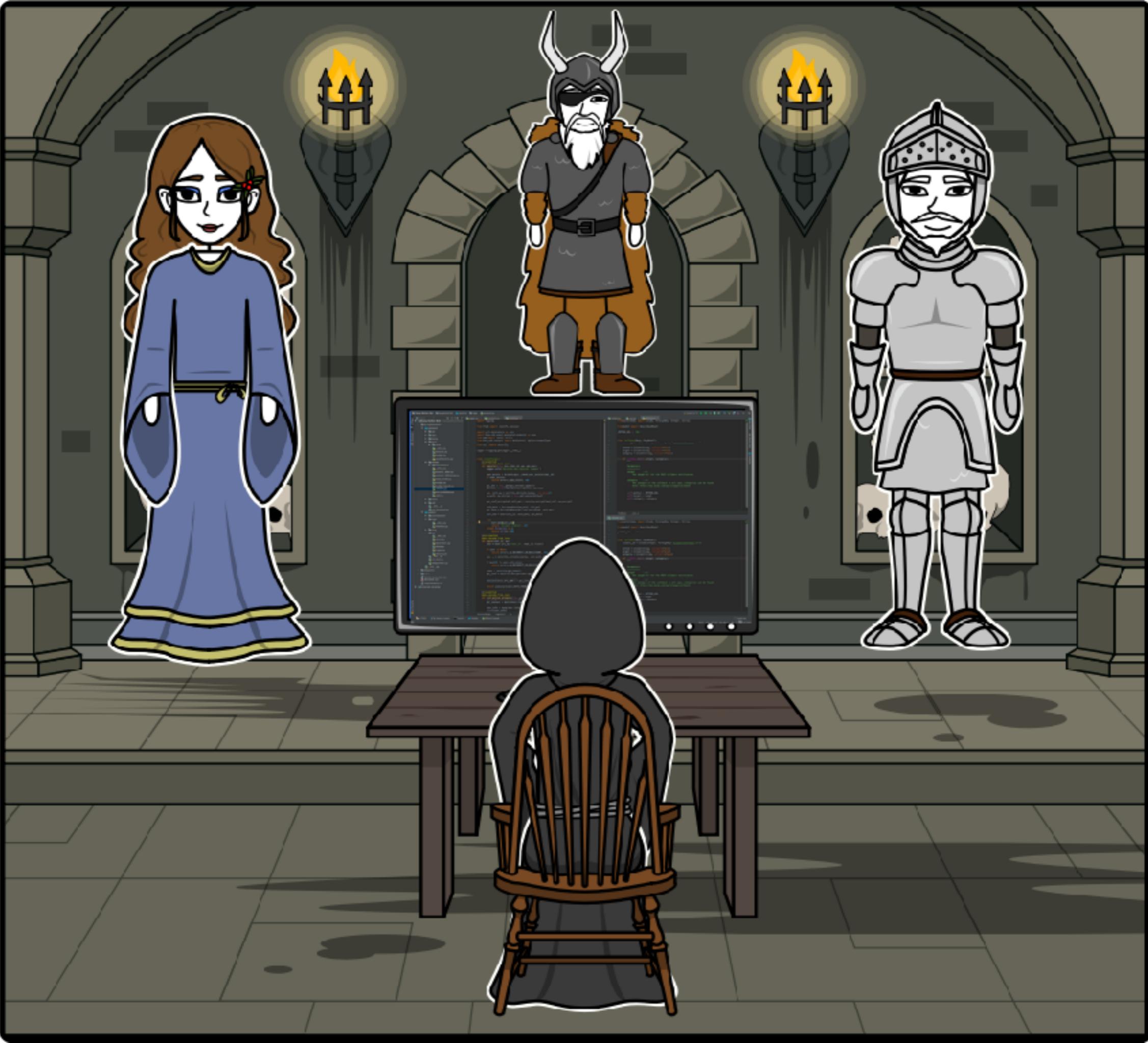
- 00001853. The current application will be terminated.
- Press any key to terminate the current application.
- Press CTRL+ALT+DEL again to restart your computer. You will lose any unsaved information in all applications.

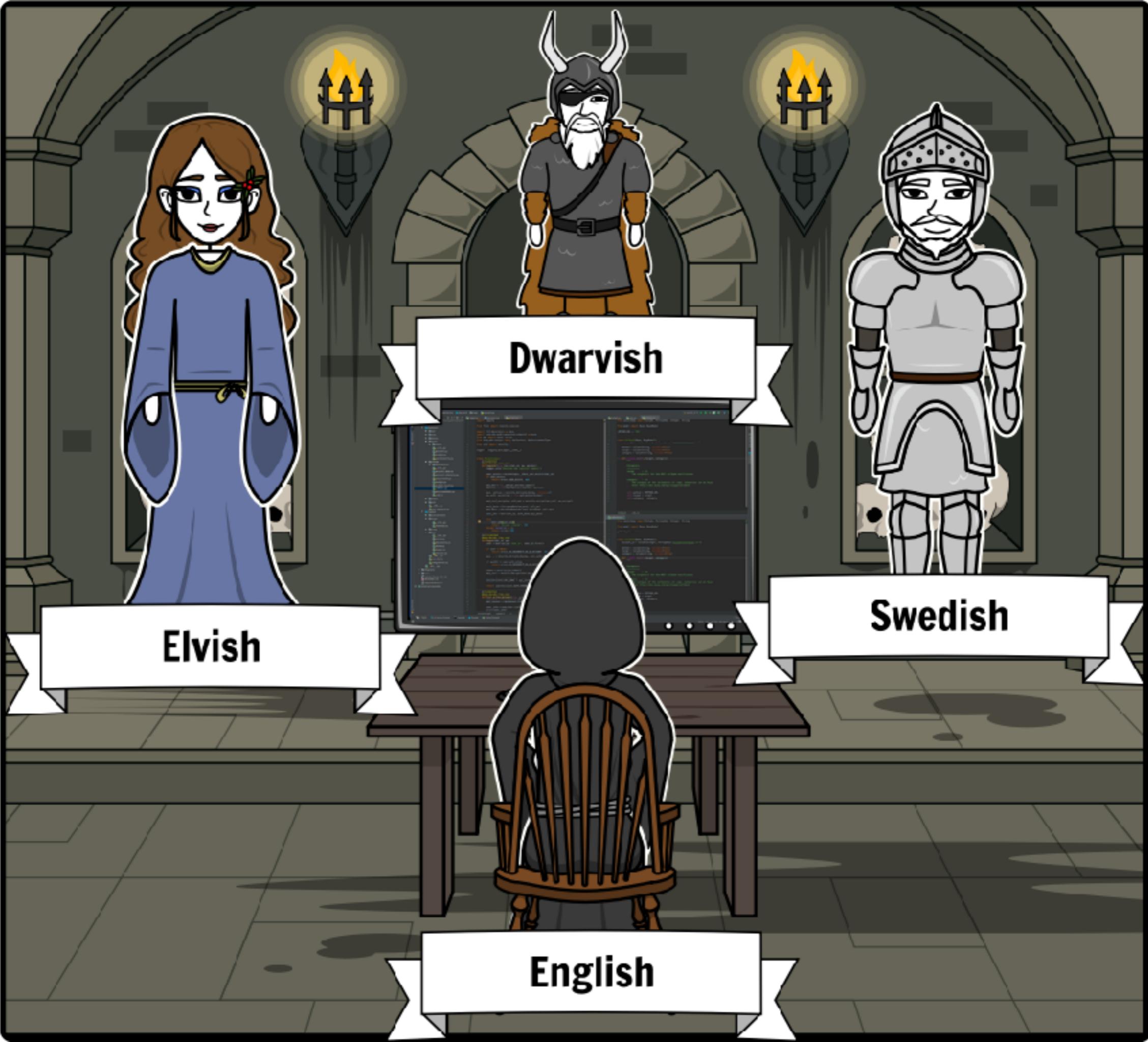
Press any key to continue _











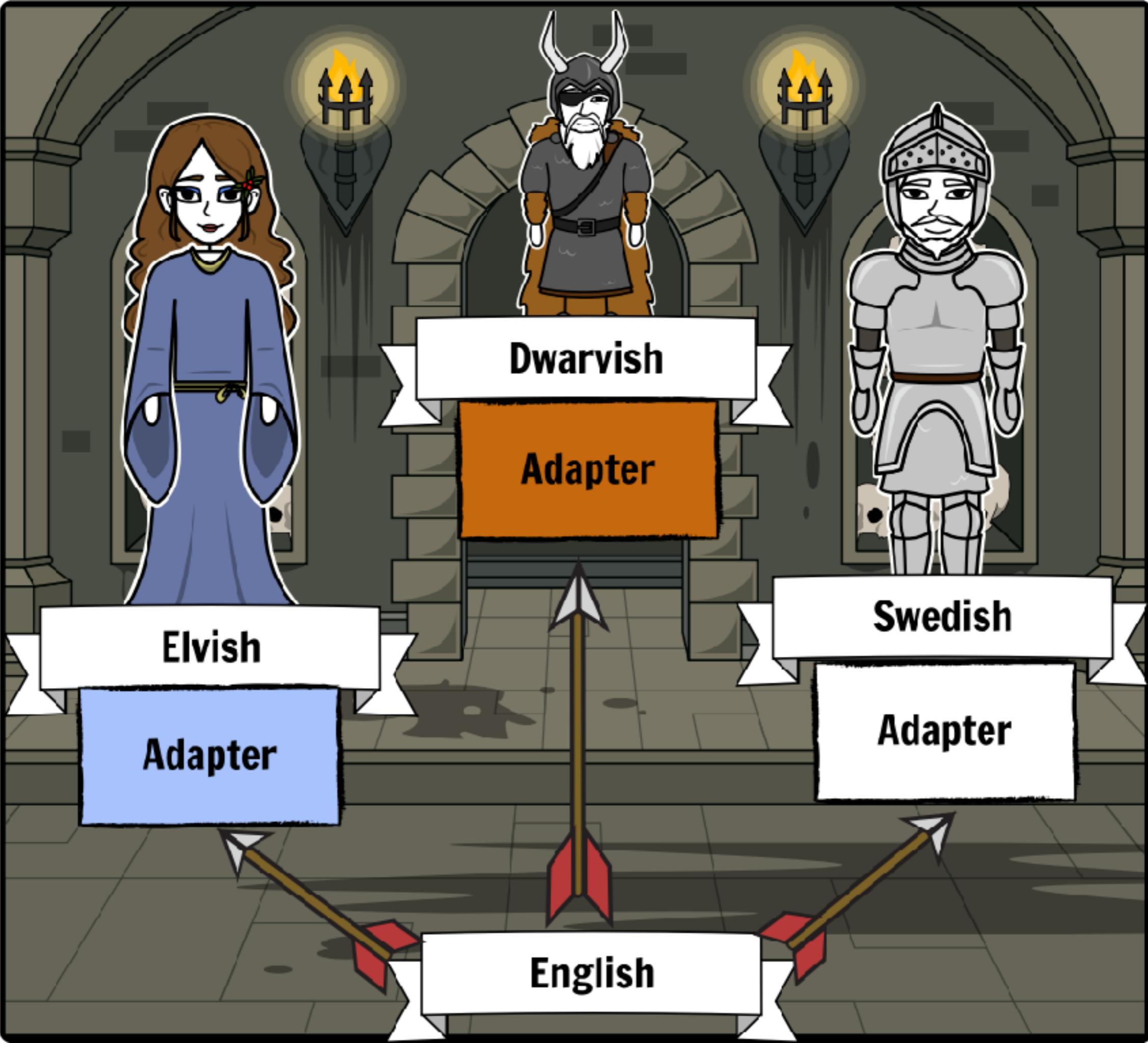
English

Elvish

Swedish

Dwarvish

ADAPTER DESIGN PATTERN

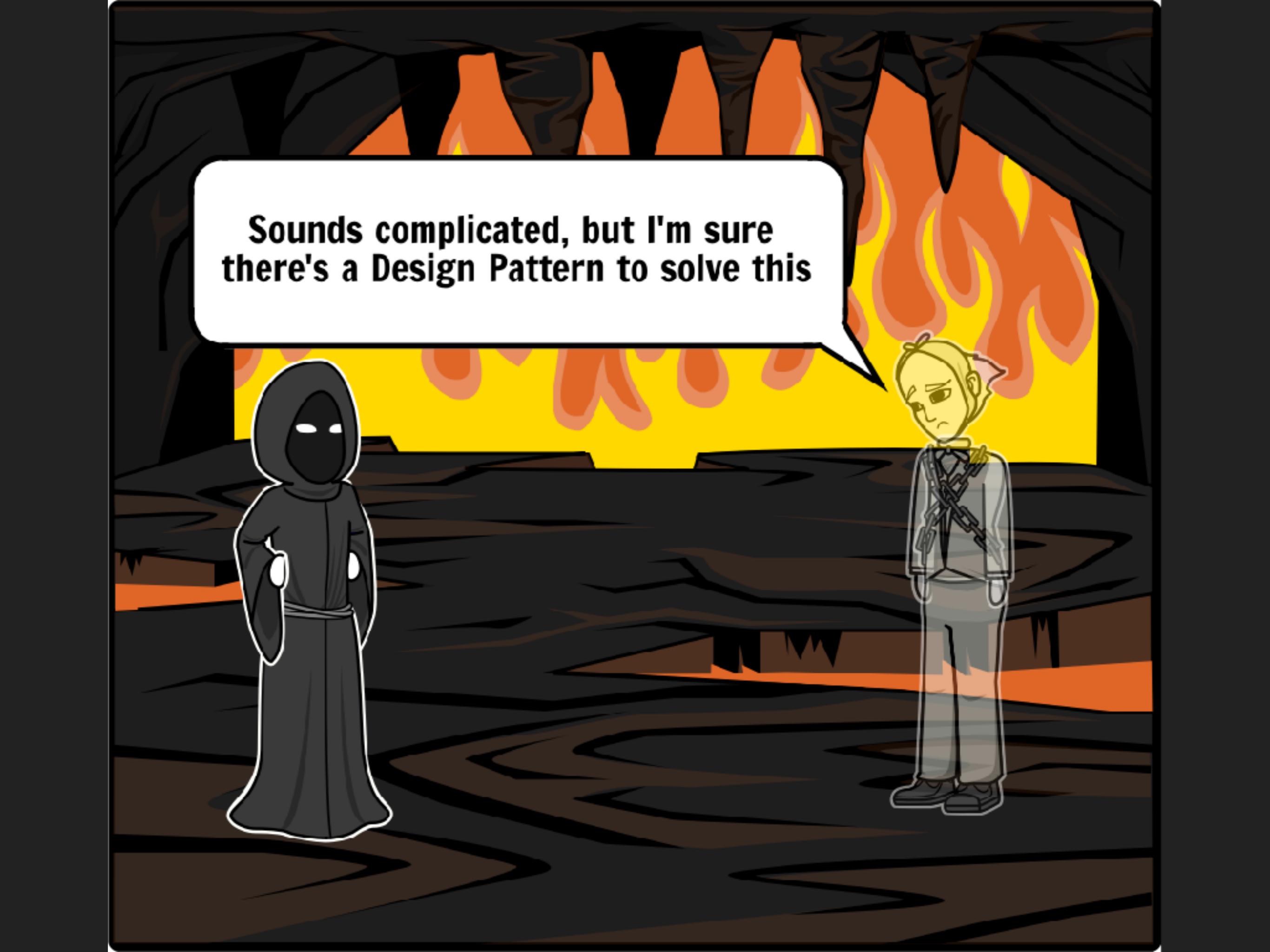


DEMO





Create ONE update
to rule them all!

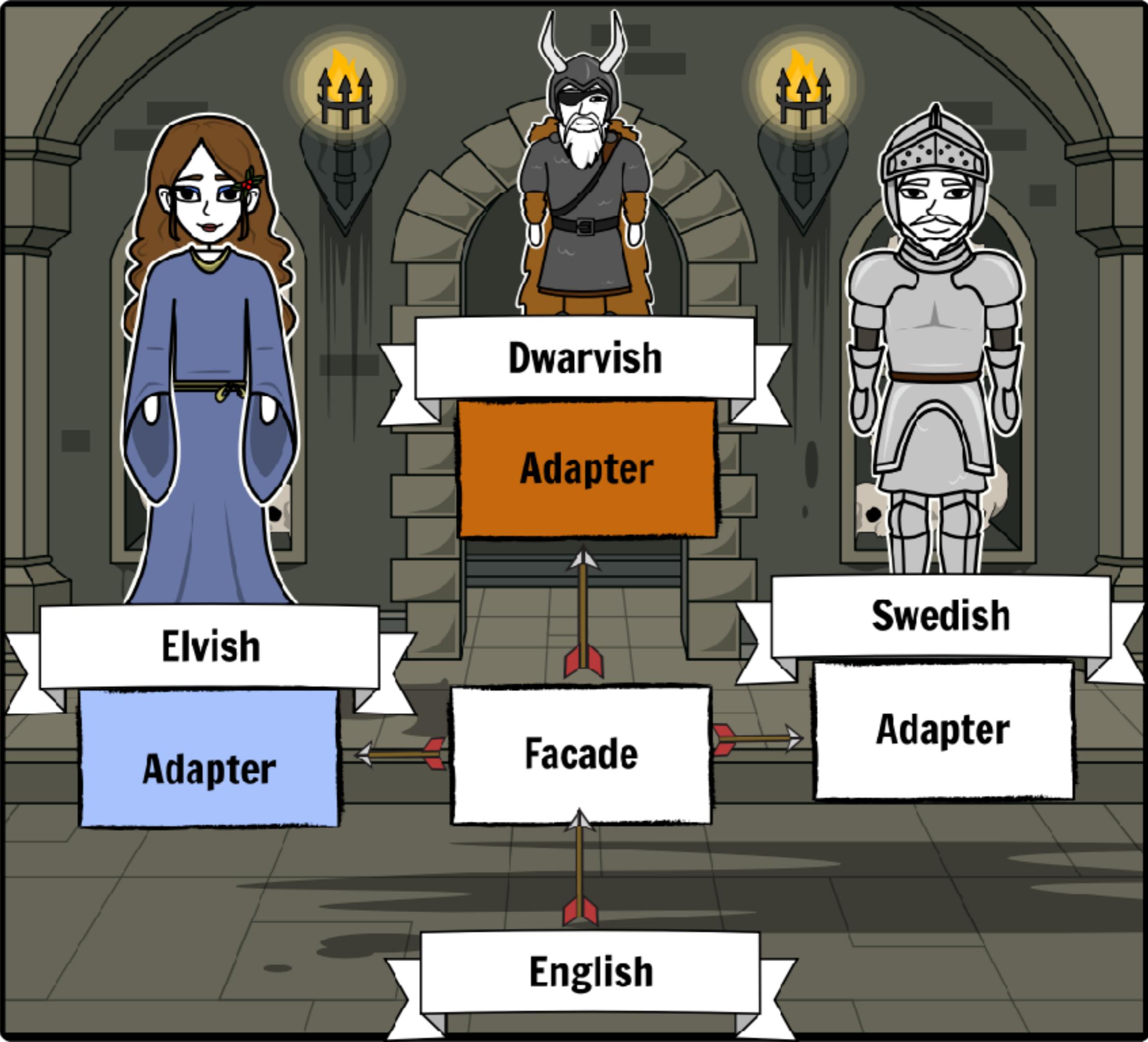


Sounds complicated, but I'm sure
there's a Design Pattern to solve this



FACADE

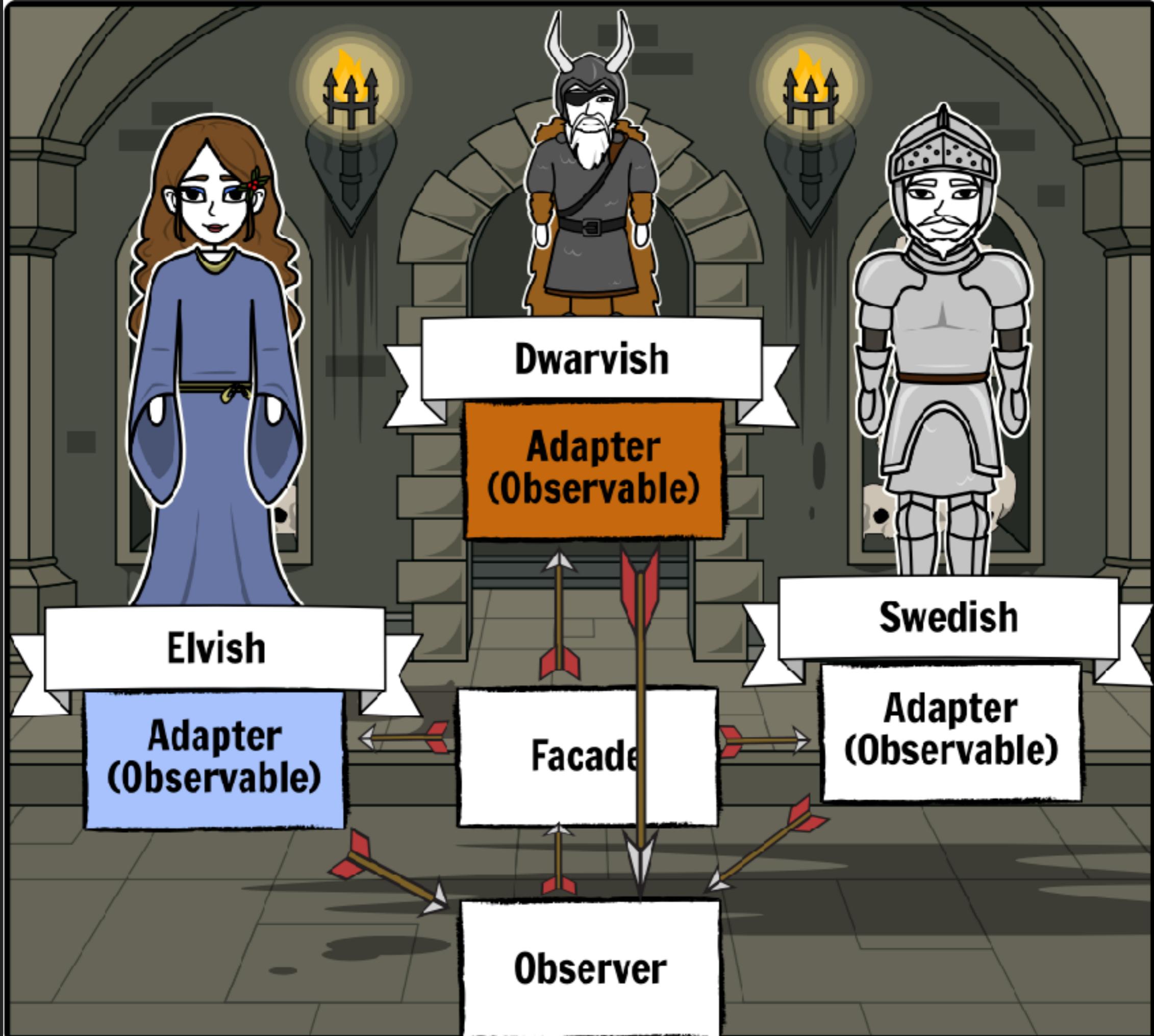
DESIGN PATTERN



DEMO



OBSERVER DESIGN PATTERN



DEMO



CRITICISM

DESIGN PATTERNS CAN BE
REPLACED WITH BUILT-IN
FUNCTIONALITY

John Doe

DESIGN PATTERNS CAN BE REPLACED WITH BUILT-IN FUNCTIONALITY

- ▶ Yes and No
- ▶ Some design patterns Yes
 - ▶ Iterator
- ▶ Many Not
 - ▶ Facade, Adapter, ...

DESIGN PATTERNS ARE MORE
FOR JAVA OR C#, NOT PYTHON

Jane Doe

DESIGN PATTERNS ARE MORE FOR JAVA OR C#

- ▶ No
- ▶ Certainly more prevalent in such fields
- ▶ But also important in Python
 - ▶ Help get a more **abstract understanding** of software
 - ▶ Help create **well structured** software in **less time**
 - ▶ Offer a **shared language** to communicate

TAKE AWAYS

TAKE AWAYS

- ▶ Have a look at Design Patterns
- ▶ Study them, Forget them, Look them up when needed
- ▶ You will write better software
- ▶ More helpful (and necessary) the more senior you become

THANK YOU

REFERENCES

- ▶ https://sourcemaking.com/design_patterns
- ▶ <http://python-3-patterns-idioms-test.readthedocs.io/en/latest/PatternConcept.html>
- ▶ <https://github.com/faif/python-patterns>