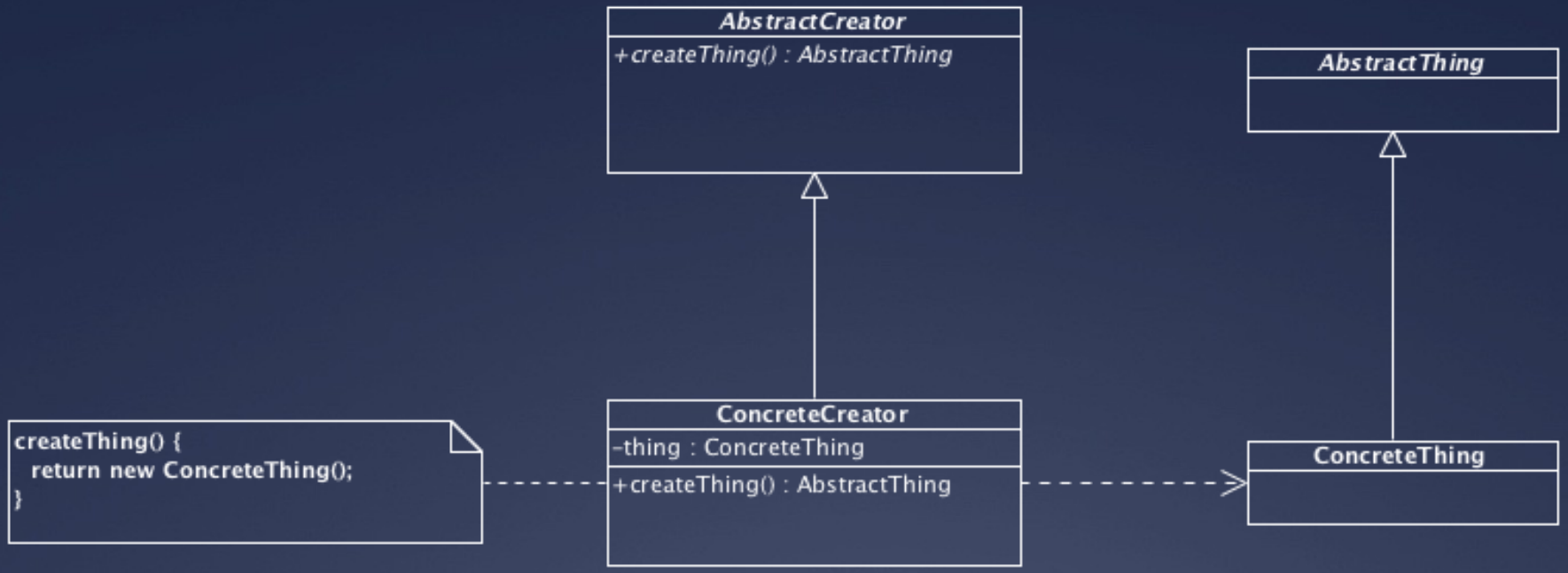
Factory Method

A Creational Design Pattern

Factory Method
Key Features

- Defines an interface for creating objects without needing to know each object's type
- Encapsulates the instantiation of concrete types
- Leaves the details of instantiation to subclasses

Factory Method UML



Factory Method Document Example

```
abstract class DocumentFactory {
  public abstract Document getDocument();
}

class HTMLCreator extends DocumentFactory {
  public Document getDocument() {
    return new HTMLDocument();
  }
}

class XMLCreator extends DocumentFactory {
  public Document getDocument() {
    return new XMLDocument();
  }
}
```

```
abstract class Document {
  ...
}

class HTMLDocument extends Document {
  ...
}

class XMLDocument extends Document {
  ...
}
```

Factory Method Encapsulation Example

```
class DocumentFactory {
  public static Document getDocument(String file) {
    int type = getType(file);
    switch(type) {
      case DocumentFactory.HTML:
        return new HTMLDocument(file);
      case DocumentFactory.XML:
        return new XMLDocument(file);
    ...
  }
}
```

Factory Method

Benefits

- Types of created objects can be determined at runtime
- Common interface allows for easier use by client classes
- Useful for toolkits and frameworks

Limitations

- Might need to subclass the factory to create concrete things (e.g., need to create an HTMLCreator to create an HTMLDocument)