APIs - what are they, really?

Web API, Programming libraries, third party APIs etc

Different kinds of APIs

Let’s consider a Java application. It uses Java interfaces and classes.

- Classes and interfaces from the standard Java API (String, List, ArrayList...)
- Classes you have written yourself (your own API)
- Classes someone else has written (third party APIs - org.json, systemet.api)

Your application might also get data from some kind of web service.

- Västtrafik API
- SMHI weather API
- Your own web api with Systembolaget products

A Web API

- Defines ways to communicate with an end-point (a web URL) in order to get som data back (or some other result)
- Typically uses HTTP GET parameters or a crafted URL with special meaning
- Typically sends data in some standard format (JSON, XML, CSV etc)
- The Servlet-system you are using in the assignments exposes data with Systembolaget products (in JSON format), via different GET parameters
- Is meant for inter-systems communication - Some system/application requests data from a very different system (in a very different location), possibly run by someone other than yourself

Your assignment - the system uses various APIs

- Your system *is in part* a web api (the servlet can send JSON according to some other system’s needs)
- Your system uses the standard Java API (classes are stored in rt.jar)
- Your system uses a Java API (systemet-api) from a JAR file you create
- Your system will use the JDBC API for SQLite3 (in lab 3)

Pretend that a separate team writes the classes for Product, ProductLine etc.

The other team packages the classes in a jar file, systemet.jar, and gives it to a team writing your web api (the servlet stuff)

Why does API stand for so many things?

We don’t know. It is stupid.

About your web API

The purpose of your web API, is to offer a way for various applications to query for data about the products in the Systembolaget product line (catalog):

- All products
- Some products
  - Specified by parameters for min_price, max_price, min_alcohol, max_alcohol
- The product data will be sent as text encoding in the JSON format
  - JSON is a notation for “objects” and arrays of objects, examples folllowing:
  - [some objects separated by comma] (an object example below):
  - { “name”: “Explorer Vodka”, “price”: 179.0, “alcohol”: 40.0, “volume”: 700}
  - (more key-value pairs can exist in a product object)

About your web API

Your web API allows other people to write applications using the product data for Systembolaget products. Since JSON is a standardized notation, such applications can be written in many different languages etc:

- Java
- Python
- PHP
- Android/Java
- iPhone Obective C
- C++
- JavaScript

About your web API

A web API is often (preferably) documented, so that developers know:

- How to query for different sets of data
- What to expect as response when doing a query

Start your Servlet and point your browser to http://localhost:8080/api.html to see an example of such documentation.

If you plan to extend the API (as a challenge), remember to update the documentation page to reflect the changes.

Java programming API

When programming in Java, you use APIs all the time. At least, you use the Standard Java API (the interfaces and classes included in your JDK installation).

These classes are compiled and put inside a JAR file (see separate lecture on JAR files) called `rt.jar` (short for run-time jar), in a place where the compiler and runtime will look and find them.

Inside the rt.jar, the whole huge directory tree (actually, many trees) with the packages you can import and use can be found.

Java programming API - rt.jar

Where your JDK (Java Development Kit) is installed, you can find the `rt.jar` in a directory called `jre/lib/` . A jar file, is very much like a normal zip file, and it contains the directory structure and class files corresponding to the complete Java standard library.

For instance, `java.util.ArrayList` can be found inside the `rt.jar` in this place:

```
java/util/ArrayList.class
```

Java programming API - rt.jar - investigated

```
$ jar tf /usr/lib/jvm/default-java/jre/lib/rt.jar | grep
ArrayList.class
com/sun/istack/internal/FinalArrayList.class
com/sun/xml/internal/messaging/saaj/util/FinalArrayList.class
com/sun/xml/internal/org/jvnet/mimepull/FinalArrayList.class
sun/swing/BakedArrayList.class
java/util/concurrent/CopyOnWriteArrayList.class
sun/awt/util/IdentityArrayList.class
java/util/Arrays$ArrayList.class
java/util/ArrayList.class
```

Listing the table of contents of a jar file

```
$ jar tf /usr/lib/jvm/default-java/jre/lib/rt.jar
```

```
t: table of contents
f: file
```

Argument(s) after **tf** should be one or more jar files

Contents will be printed to standard out (e.g. your terminal)

What other APIs can you use when programming?

Apart from the rt.jar standard Java library classes, you can program against classes and interfaces in other JAR files.

If you write a Servlet, you must import and extend the HttpServlet class:

```
$ grep -i http webroot/WEB-INF/classes/servlets/SystemetWebAPI.java
import javax.servlet.http.*;
public class SystemetWebAPI extends HttpServlet{
  public void doGet(HttpServletRequest request, HttpServletResponse response)
```

Those classes are not included in rt.jar, so you need to put e.g. winstone.jar (or some other implementation of those packages, interfaces and classes) on your classpath.

What other APIs can you use when programming?

Your web API deals a lot with classes representing stuff about the Systembolaget products:

```
se.itu.systemet.domain.Product
se.itu.systemet.storage.ProductLineFactory
se.itu.systemet.storage.ProductLine
```

We decidet to let you develop (finish our code) those classes and interfaces separately from the web API.

The Servlet classes needs to import and use them, though. So how to do that?

What other APIs can you use when programming?

We decided to show you that you can package the whole `se/` directory structure (including `se/itu/systemet/{domain,storage}/`) in a JAR file, and put that JAR file where the Servlet (and compiler) can find it.

This is the task of the `deploy_systemet_jar.sh` script - it finds and compiles all classes under `se/` and puts them inside a jar file, `systemet.jar` and finally puts that JAR file in a place where the Servlet will find it: `webroot/WEB-INF/lib/` .

Just like it's a standard to put `rt.jar` in `jre/lib/` it is a standard to put external jar files in `WEB-INF/lib/` for Servlets.

What’s the point and implication of systemet.jar?

- The servlet code can be developed separately from systemet api classes
- You can ask a different team for the systemet.jar and use their implementation of all the product related classes
    - The systemet-api becomes replaceable
- The second team needs only to send you one file (systemet.jar) instead of a whole directory structure with all classes and packages

Try for yourselves! Use some other person’s or team’s systemet.jar

Other Library APIs in the assignment

In Lab 3 where you are integrating all the parts (your GUI talks to the Servlet, the Servlet talks to a database, and your GUI converts the JSON from the Servlet back to Java objects), you will use some additional JAR files with APIs:

- sqlite.jar - An implementation of JDBC which let's you write Java code to talk to an SQLite database
- org.json.jar - A library for creating and parsing JSON from and to Java

Further reading

- https://docs.oracle.com/javase/8/docs/technotes/tools/windows/findingclasses.html
- https://docs.oracle.com/javase/8/docs/technotes/tools/windows/classpath.html
- https://docs.oracle.com/javase/8/docs/technotes/tools/windows/jdkfiles.html
- https://docs.oracle.com/javase/8/docs/technotes/tools/windows/jar.html
- https://en.wikipedia.org/wiki/Application_programming_interface#Libraries_and_frameworks
- https://en.wikipedia.org/wiki/Application_programming_interface#Web_APIs
- https://en.wikipedia.org/wiki/Web_API