ECE 448/528 Application Software Design

Lecture 5. Client/Server Model

Spring 2025

Won-Jae Yi, Ph.D.

Department of Electrical and Computer Engineering
Illinois Institute of Technology

Client/Server Model

Client/Server Model

- A network programming model for networked applications.
  - Mostly based on TCP for their close relationship.
  - A server program runs on the server.
  - A client program used by the clients.
- A server is a high-performance computer designed to store and manage data online.
- The clients connect to the server to utilize the computational resources and data storage.
- The server may additionally enable and facilitate communication between clients.
- Highly-available (HA) server farms may allow clients to remain connected even if some servers fail.

Client/Server Application Example

- A server helps clients to reverse strings (as an example).
    - The clients and the server communicate via the TCP protocol.
- Is this requirement clear?
- Questions to be answered.
    - What is a string?
    - How does a client send one string to the server?
    - How does a client send multiple strings to the server?
    - Can the server handle multiple clients at the same time?
- Can we decompose the application into testable pieces?

Strings and Bytes

User Story

- A widely used tool for requirements analysis.
- One of the important factors in Agile Software Development
  - Divide 'big' development pieces into 'stories' or 'small/short' development pieces
- Describe one or more desired features
  - Use a <u>short and informal</u> piece of natural language.
  - For both end users and developers.
- "As a ... (user/who), I want ... (action/what), so that ... (purpose/why)."
  - SMART: Specific, Measurable, Achievable, Relevant, Time-bound
- Example: As a developer, I want to reverse a Java String using `ReverseString.reverse`, so that I can use this function to build the application.

Step 1: Red

```
package ece448.lec05;
...
public class ReverseStringTests {
  @Test
  public void test() {
    assertEquals(ReverseString.reverse("Hello"), "olleH");
  }
}
```

- `src/test/ece448/lec05/ReverseStringTests.java`
- Red: it will not pass now
- Annotations are widely used in Java to communicate intentions in a declarative but not imperative way.
  - The actual purpose depends on the library providing it.

Step 1: Red (cont.)

```
package ece448.lec05;
public class ReverseString {
   public static String reverse(String str) {
      return "";
   }
}
```

- `src/main/ece448/lec05/ReverseString.java`
- A simple method definition without actual implementation.
- Red: the code compiles but the test will not pass now
- `static` keyword is a non-access modifier used for methods and attributes. Static methods/attributes can be accessed without creating an object of a class.
- `public` classes, methods, and variables can be accessed by any other part of the Java program.

Step 2: Green

```
package ece448.lec05;
...
public class ReverseString {
  public static String reverse(String str) { StringBuilder
    sb = new StringBuilder(str);  return
    sb.reverse().toString();
  }
}
```

- You may use a loop or a library function.
- Green: the test will pass now.
- More test cases?

Internationalization (i18n)

- Your software should work with **all-natural languages**.
    - Not just for ASCII or what your OS could handle.
    - a.k.a. i18n and l10n (internationalization and localization).
- Strings are NOT simply arrays of bytes.
    - There are more characters than what an 8-bit byte can hold.
    - A Java `String` holds an array of Java `char`, which is 16-bit.
- Unicode: standards for handling text around the world.
    - Code point: “character” of Unicode, most are actual characters but could also be for formatting purposes.
    - Code unit: underlying units for binary encoding.
- What is Java `String` and Java `char`?
    - Originally UCS-2: 16-bit unit, 65,536 code points only.
    - Currently UTF-16: variable-length of one or two 16-bit units.
    - `char` is always the code unit.
    - So, `String.length` returns the number of code units but not code points – mostly this is OK but watch out for special cases.

Unicode as Bytes

- While you may encode a 16-bit `char` as 2 bytes, Unicode does include standards regarding conversion from/to bytes.
  - For things like files and TCP streams that work with bytes.
  - Used in the presentation layer for computer networking.
- UTF-8: variable-length of one to four 8-bit units.
  - UTF (Unicode Transformation Format)
  - More compact than UTF-16.
  - Fall-back to ASCII strings if all characters are within 7-bit ASCII characters.
- Use `String(byte[], "UTF-8")` to convert from bytes.
- `String.getBytes("UTF-8")` to convert to bytes.