HTML

HTML is a *markup language*, meaning that you add tags to specify how text is to be handled.

```
<html>
<head>                          page title
<title>Hello World!</title>     specifies information
</head>                         about the document
<body>                          section heading
<h1>A Heading</h1>              content
<p>Hello world, it is so nice to meet you.</p>   to be
</body>                         displayed
</html>                         paragraph
```

- tags primarily address document structure
  – headings, paragraphs, tables, links, images, ...
- some tags address formatting, though CSS is preferred
  – bold, italics, fonts, ...

PHP

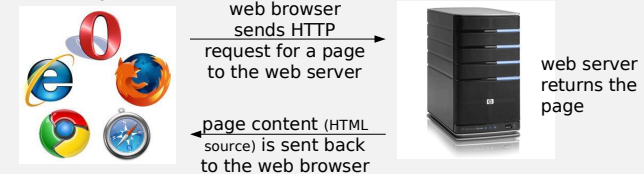PHP is a programming language.

- basic concepts (variables, if statements, loops, arrays, comments, etc) are the same as in Java and many other languages
  – much of the syntax is similar, too

- has object-oriented features, but we won't be using them

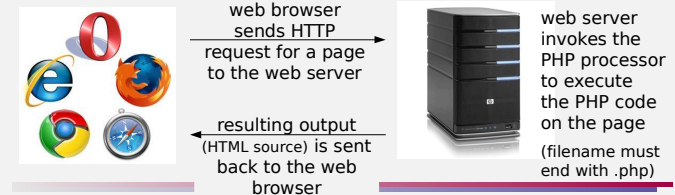Our goal is to use PHP to generate web pages in order to have web pages with dynamic content.
  – the PHP program's output is HTML source

PHP on Web Pages

- PHP is embedded within special tags in a file

```
<?php
  …
?>
```

- everything outside the <?php … ?> tags is output literally
  (convenient to avoid having to write lots of print statements and manage "" when there is static content on the page)

Take care for readability – make good use of whitespace!

PHP on Web Pages

HTML only

web browser sends HTTP request for a page to the web server

web server returns the page

page content (HTML source) is sent back to the web browser

HTML with PHP

web browser sends HTTP request for a page to the web server

web server invokes the PHP processor to execute the PHP code on the page

(filename must end with .php)

resulting output (HTML source) is sent back to the web browser

HTML Forms

HTML forms provide a way to get user input.

default text	⊙ yes
default text	○ no
	○ maybe
New York ▼	☑ yes
	☐ no

[reset form] [submit form]

When a form is submitted, a request containing the input values is sent to the web server.
- can access the values through certain variables in the PHP environment

Maintaining State

HTTP is a stateless protocol.

To store information between requests, another mechanism is needed.
- hidden form elements
  - simple, HTML-based
  - values are publicly visible in the page source
  - last for only one request
  - requires fixed, linear sequence of pages
- cookies
  - stored by user's web browser
  - last until they expire, even if browser is restarted
  - users may disable
- sessions
  - PHP-based
  - last until browser exits (or might timeout)

Sending Queries – Simple Queries

- suitable for one-time queries without user input

```
$query = "SELECT S.Sid,S.Sname,B.Bid,B.Bname,R.Day FROM BOAT B ".
         "NATURAL JOIN RESERVATION R NATURAL JOIN SAILOR S ".
         "ORDER BY S.Sid ASC, B.Bid ASC,R.Day ASC";
$result = mysqli_query($link,$query);
```

```
Statement stmt = connection.createStatement();
ResultSet result = stmt.executeQuery("SELECT * FROM BOOK NATURAL JOIN BOOK_AUTHORS");
```

Sending Queries – Prepared Statements

- suitable for…
  - queries incorporating user input
  - suitable for repeated similar queries

- why prepared statements?
  - protects against SQL injection when user input is utilized for values in the query
  - provides greater efficiency if query is used multiple times with different values

Prepared Statements

- steps
  - define query text with placeholders for values and send to the DB for parsing
  - set values for the placeholders
  - execute the query   } repeat as desired

```
$query = "SELECT B.Bname,B.Bid,R.Day FROM BOAT B NATURAL JOIN ".
         "RESERVATION R NATURAL JOIN SAILOR S WHERE S.Sname=?";
$stmt = mysqli_prepare($link,$query);

mysqli_stmt_bind_param($stmt,'s',$_POST["sailor"]);

mysqli_stmt_execute($stmt);
```

```
PreparedStatement stmt = connection
            .prepareStatement("SELECT * FROM BOOK NATURAL JOIN BOOK_AUTHORS WHERE Author_name=?");
stmt.setString(1, "C. J. Cherryh");
ResultSet result = stmt.executeQuery();
```