Functions: eval and show_source

- eval →evaluates string as PHP code
- **Example:** EvalFunction.php
- Applications (examples)
  - Generate code in a loop and execute afterwards
  - Store code (e.g., a database or file) and later execute
- show_source (alias for function highlight_file) → syntax highlighter
- **Example:** Show.php

File Uploading

- You can upload files to a specific directory in the web server
- We rely on the $_FILES superglobal to obtain information about the uploaded file
- Information
  - Name → associated with “name”
  - Temporary name → associated with “tmp_name”
  - File size → associated with “size”
  - File Type → associated with “type”
- Support functions
  - is_uploaded_file → indicates whether file was uploaded
  - move_uploaded_file → allow us to copy file from temporary location
- To run the following file upload examples
  - You can use the files available in the code distribution “FileUploadExampleFiles” folder or you can use your own files (you need a pdf file, a text file and an image file). If using the code distribution files copy them to your Desktop (to make the example more realistic)
  - Create a directory in your computer (e.g., C:\tempExample) where uploaded files will be placed
- **Example:** fileUpload.html, fileUpload.php
- Notice enctype="multipart/form-data“ in the form
- **Example**: multipleFileUpload.html, multipleFileUpload.php

Accessing Individual Characters

- You can use {c} to access characters in a string.
- **Example:** Character.php

Additional String Functions

- **Example:** AdditionalStringFunctions.php
- **printf** → function that outputs data to the browser
- Requires a string called the format control string. Within the formal control string you can have a conversion specification (which begins with %)
- Several conversion specifiers

  %d → display as decimal

  %c → display integer as character

  %f → display as floating-point number (double)

  %s → display as string

  %x → display as hexadecimal
- **Example:** printf.php
- **sprintf** → extremely useful

Formatting Numbers

- We can use number_format for number formatting
- **Example:** Format.php

Useful Array Functions

- in_array → returns true if array contains specified element
- shuffle → randomizes the elements in the array
- implode → turns array into string
- explode → turns string into array
- array_merge → combines two or more arrays
- **Example:** Arrays.php

Input Validation

- Always verify that a particular element exists before applying any other validation process
- To test if a value has been provided in a text box use strlen and not empty
- **Example:** Validation.php
- Checking the type of a variable via
  - is_array()
  - is_bool()
  - is_float()
  - is_int()
  - is_null()
  - is_numeric() → (integer or float)
  - is_object()
  - is_string()

Input Validation

- Assuming $x is a string (trimmed) and not empty string
- Integer >= 0

  if (!ctype_digit($x)) { echo “INVALID” }
- Positive or negative integer

  if ($x != strval(intval($x))) { echo “INVALID” }
- Positive or negative decimal

  if ($x != strval(floatval($x))) { echo “INVALID” }
- intval/floatval → ignore initial whitespace and returns as much number as it can find in the string
- **Example:** ValidationNumbers.php

Functions with Variable Parameter Count

- Variable length parameter list is possible in php
- **Example:** VariableParameterFunction.html
- You can also use func_get_args() to return an array of the arguments that were passed in

Creating Aliases

- We can use & to create variable aliases
- **Example:** Aliases.php

Buffering

- ob_start → creates output buffer
- ob_end_flush → ends the buffer and sends the output
- ob_end_clean → ends the buffer without sending the output
- **Example:** Buffering.php
- Output buffering enables us to compress the HTML you send to the browser
- If a browser supports compressed HTML it lets the server know on each page requests
- Compressed HTML → zipped version of HTML

PEAR/PECL

- **PEAR** (PHP Extension and Application Repository)
  - Only a few of PEAR packages are part of the PHP release
  - Web site: http://pear.php.net
  - pear program in PHP release allows you to download and install additional packages
  - Instructions and examples for a particular package are available at http://pear.php.net/packages.php
- **PECL** (PHP Extension Community Library)
  - Extensions to PHP written in C
  - Similar to the ones distributed with the PHP release
  - Examples: interface to libssh2 library, graphics library
  - https://pecl.php.net/

LAMP

- Collection of software
  - Used to run dynamic Web Sites
  - Usually free/open-source
- LAMP
  - L → Linux
  - A → Apache
  - M → MySQL
  - P → Perl, Python, PHP
- Site of Interest
  - http://www.onlamp.com/

Encryption

- **Encryption** → process of converting plaintext into ciphertext.
- **Decryption** → process of converting ciphertext into plaintext
- **Symmetric cryptography** → sender and receiver share the same key
- **Asymmetric (Public Key) cryptography** → sender and receiver have different, complementary keys
- **Symmetric cryptography**
  - Example algorithms: DES, Triple-Des, RC4
  - Relatively fast compared to Asymmetric
  - Drawbacks
    - Keys must be changed frequently
    - How to distribute the key safely

PHP's crypt function

- crypt → returns an encrypted string using Unix DES-based encryption or alternative algorithms available in the system
- Arguments
  - String to encrypt
  - Optional salt string to base the encryption on (if none is provided one is randomly generated by PHP each time the function is called)
- On systems where multiple encryption types are available the following constants are set (0 or 1) to indicate whether the encryption type is available
  - CRYPT_STD_DES → DES-based encryption
  - CRYPT_MD5 → MD5 encryption
  - CRYPT_EXT_DES → Extended DES-based encryption
  - CRYPT_BLOWFISH → Blowfish encryption
- **Example: EncryptionWithCrypt.php**

Checksum Generation

- Checksum (Hash or Message Digest)
  - String that allow us to verify the correctness of data
  - It is easy to determine that a particular string (data we want to verify) matches the checksum
  - It is not (easily) possible to re-create the source string from the checksum
- Expected properties for message digest ("Hashing") algorithm
  - Original message cannot be obtained from the digest
  - Two different messages should have different digests
- Two algorithms for checksum generation
  - SHA1 → Secure Hash Algorithm
  - MD5 → Message Digest Algorithm
- DES/crypt vs (MD5 and SHA1)
  - MD5 and SHA1 checksum for a string is always the same
- **Example: EncryptionWithSHAMD5.php**
- Checksums are used to validate downloads
- md5_file and sha1_file functions
  - Allow us to generate the hashes of a file by opening and reading the data

LDAP

- Lightweight Directory Access Protocol
  - Directory → Collection of objects organized in logical and hierarchical structure
    - Example: Telephone Directory
- University of Maryland Directory uses LDAP
  - Faculty, staff, and students and others have records in the University directory
  - A record contains standard office and telephone information, userID and password verification
  - http://directory.umd.edu/
- Windows user (XAMPP)
  1. uncomment *extension=php_ldap.dll* in \xampp\php\php.ini
  2. Copy \xamp\php\libsasl.dll to \xampp\apache\bin
  3. Remember to restart the server
- **Example:** ldapExample folder