Error and Exception Handling in PHP
• Exception handling is used to change the normal flow of the code execution if a specified error (exceptional) condition occurs. This condition is called an exception.
• The current code state is saved.
• The code execution will switch to a predefined (custom) exception handler function.
• Depending on the situation, the handler may then resume the execution from the saved code state, terminate the script execution or continue the script from a different location in the code.
Basic Use of Exceptions

• When an exception is thrown, the code following it will not be executed, and PHP will try to find the matching "catch" block.

• If an exception is not caught, a fatal error will be issued with an "Uncaught Exception" message.
Basic Use of Exceptions

• Lets try to throw an exception without catching it:

```php
<?php

//create function with an exception
function checkNum($number) {
    if($number>1) {
        throw new Exception("Value must be 1 or below");}
    return true;}

//trigger exception
checkNum(2);

?>
```
The following message will be thrown:

**Fatal error: Uncaught exception 'Exception' with message 'Value must be 1 or below' in C:\xampp\htdocs\ibm.php:5**

Stack trace:

#0 C:\xampp\htdocs\ibm.php(8): checkNum(2)
#1 {main}

thrown in C:\xampp\htdocs\ibm.php on line 5
Try, Throw and Catch

• To avoid the error from the example above, we need to create the proper code to handle an exception.

• Proper exception code should include:
  Try - A function using an exception should be in a "try" block. If the exception does not trigger, the code will continue as normal.

• However if the exception triggers, an exception is "thrown"
Try, Throw and Catch

- Throw - This is how you trigger an exception.
- Each "throw" must have at least one "catch"
- Catch - A "catch" block retrieves an exception and creates an object containing the exception information
Try, Throw and Catch

• Lets try to trigger an exception with valid code:

```php
//create function with an exception
function checkNum($number) {
    if($number>1) {
        throw new Exception("Value must be 1 or below");
    }
    return true;
}
//trigger exception
checkNum(2);`
Try, throw and catch

//If the exception is thrown, this text will not be shown
echo 'If you see this, the number is 1 or below';}

//catch exception

catch(Exception $e)
{
    echo 'Message: ' . $e->getMessage();
}

?>

Output-

Message: Value must be 1 or below
Custom Exception

• Creating a custom exception handler is quite simple.
• We simply create a special class with functions that can be called when an exception occurs in PHP.
• The class must be an extension of the exception class.
• The custom exception class inherits the properties from PHP's exception class and you can add custom functions to it.
Custom Exception

- Lets create an exception class:

```php
<?php

Class NegativeageException extends Exception
{
    private $info;
    public function __construct($message)
    {
        $this->info=$message;
    }
    public function getInfo()
    {
        return $this->info;
    }
}
```
Custom Exception

$stuname="Deepak";
$age=-23;
$mark=34;
try {
    if($age<=0) {
        throw new NegativeageException("Age can’t be negative");
    }
}
catch(NegativeageException $n)
{
    echo $n->getInfo();
}
Multiple Exceptions

• It is possible for a script to use multiple exceptions to check for multiple conditions.
• It is possible to use several if..else blocks, a switch, or nest multiple exceptions.
Exception Propagation

• When an exception occurs and if it is not caught by any of the catch blocks, it looks for its handler and if it fails to find one it just propagates from one method to other and ends up crashing the program.
Rules for Exceptions

• Code may be surrounded in a try block, to help catch potential exceptions.
• Each try block or "throw" must have at least one corresponding catch block.
• Multiple catch blocks can be used to catch different classes of exceptions.
• Exceptions can be thrown (or re-thrown) in a catch within a try block.
• A simple rule: If you throw something, you have to catch it.