

I DID NOT DEVELOP THE PHP
WE KNOW TODAY. DOZENS, IF
NOT HUNDREDS OF PEOPLE
DEVELOPED PHP. I WAS SIMPLY
THE FIRST DEVELOPER.

—RASMUS LERDORF—



Introduction to PHP

PHP හැඳින්වීම

PHP is a server-side scripting language that is widely used for web development.

PHP යනු වෙබ් අඩවි සංවර්ධනය සඳහා බහුලව භාවිතා වන සේවාදායක අග්‍රයෙහි ක්‍රියාත්මක වන ස්ක්‍රිප්ට් කාණ්ඩයකි.

PHP, originally known as **Personal Home Page**, was created by Rasmus Lerdorf in 1994.

ආරම්භයේදී Personal Home Page ලෙස හැඳින්වූ PHP , 1994 දී Rasmus Lerdorf විසින් නිර්මාණය කරන ලදී.

Initially, it consisted of Perl scripts designed to manage his personal website and track visits.

මුලදී, එය ඔහුගේ පුද්ගලික වෙබ් අඩවිය කළමනාකරණය කිරීම සහ පැමිණීමේ නිරීක්ෂණය කිරීම සඳහා නිර්මාණය කරන ලද නාමාවේන් ලියූ ස්ක්‍රිප්ට් වලින් සමන්විත විය.

Recognizing its potential beyond personal use, Lerdorf rewrote these tools in C language and released PHP/FI (Forms Interpreter) 1.0 in 1995.

පුද්ගලික භාවිතයෙන් ඔබ්බට එහි විභවය හඳුනාගෙන, Lerdorf විසින් මෙම මෙවලම් C භාෂාවෙන් නැවත ලියා 1995 දී PHP/FI (Forms Interpreter) 1.0 ලෙසින් නිකුත් කළේය.

It provided basic tools for web form processing and database communication.

එය වෙබ් ආකෘති සැකසීම සහ දත්ත සමුදා සමග සන්නිවේදනය සඳහා මූලික මෙවලම් සපයන ලදී.

Later versions

PHP 3.0 (1998)

Introduced a modular architecture and expanded database support.

මොඩියුලර් නිර්මිතියක් සහිත සහ පුළුල් දත්ත සමුදා සහායන් හඳුන්වා දෙන ලදී.

PHP 4.0 (2000)

Powered by the Zend Engine This version brought significant performance improvements and advanced features like sessions and output buffering.

Zend Engine මගින් බලගන්වන ලද මෙම අනුවාදය සැලකිය යුතු කාර්ය සාධන වැඩිදියුණු කිරීම් සහ සැසි සහ ප්‍රතිදාන වැනි උසස් විශේෂාංග ගෙන වලින් සමන්විත විය.

PHP 5.0 (2004)

Enhanced support for object-oriented programming (OOP) with the introduction of classes, interfaces, exceptions, and improved XML handling.

classes, interfaces, exceptions, සහ වැඩිදියුණු කළ XML හැසිරවීම හඳුන්වාදීම සමඟ වස්තු-නැඹුරු ක්‍රමලේඛකරණය OOP සඳහා වැඩි දියුණු කළ සහාය.

PHP 7.0 (2015)

Delivered major performance enhancements with Zend Engine 3.0, adding scalar type declarations, return type declarations, and better error handling.

Zend Engine 3.0 සමඟින් ප්‍රධාන කාර්ය සාධන වැඩි දියුණු කිරීම් ලබා දී, අදිශ ආකාරයේ ප්‍රකාශ වකතු කිරීම, return type ප්‍රකාශයන් සහ වඩා හොඳ දෝෂ හැසිරවීමක් ගෙන දුන්නේය.

PHP 8.0 (2020)

Introduced the JIT compiler for further performance gains, along with new features like union types, named arguments, and match expressions.

union types, named arguments, සහ match ප්‍රකාශන වැනි නව විශේෂාංග සමඟින් තවදුරටත් කාර්ය සාධනයක් සඳහා (අක්ෂර) සම්පාදකය හඳුන්වා දෙන ලදී.

Static webpages

ස්ථිතික වෙබ් පිටු

Static webpages display fixed content. They are built using HTML and sometimes CSS and JavaScript.

ස්ථිතික වෙබ් පිටු ස්ථාවර අන්තර්ගතයන් පෙන්වයි. ඒවා HTML සහ සමහර අවස්ථා වලදී CSS සහ JavaScript භාවිතයෙන් ගොඩනගා ඇත.

The content remains the same for every user and doesn't change unless the web developer manually updates the HTML code.

සෑම පරිශීලකයෙකුටම අන්තර්ගතය එකම ලෙස පවතින අතර වෙබ් සංවර්ධකයා විසින් HTML කේතය අතින් යාවත්කාලීන කරන්නේ නම් මිස වෙනස් නොවේ.

Dynamic webpages





ගතික වෙබ් පිටු

Dynamic webpages display content that can change based on user interactions, preferences, or other factors.

ගතික වෙබ් පිටු පරිශීලක අන්තර්ක්‍රියා, කැමැත්ත, හෝ වෙනත් සාධක මත පදනම්ව වෙනස් විය හැකි අන්තර්ගත සංදර්ශන කරයි.

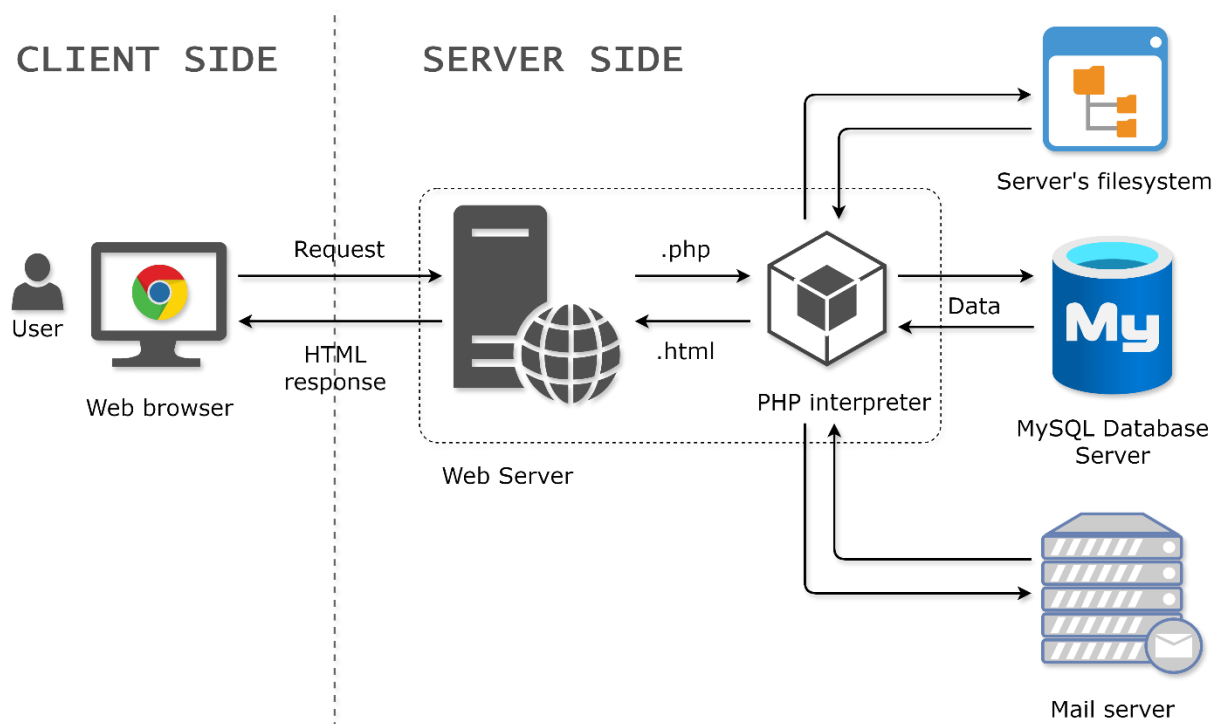
They are created using server-side scripting languages like PHP, Ruby, Python, or Node.js, and often interact with databases to retrieve and store data.

ඒවා PHP, Ruby, Python, හෝ Node.js වැනි සේවාදායක අග්‍රයෙහි ක්‍රියාත්මක වන ස්ක්‍රිප්ට් භාෂා භාවිතයෙන් නිර්මාණය කර ඇති අතර, දත්ත ලබා ගැනීමට සහ ගබඩා කිරීමට බොහෝ විට දත්ත සමුදායන් සමඟ අන්තර් ක්‍රියා කරයි.

Feature	Static Webpages	Dynamic Webpages
Content අන්තර්ගතය	Fixed ස්ථිතිකය	Changes based on user interaction පරිශීලක අන්තර්ක්‍රියා මත පදනම් වී වෙනස් වෙයි
Generation නිර්මාණය	Pre-generated HTML පෙර-ජනනය කරන ලද HTML	Generated with server-side scripts සේවාදායක අග්‍රයෙහි භාෂා මගින් ජනනය කර ඇත
Complexity සංකීර්ණත්වය	Simple (HTML, CSS) සරලයි	Complex (server-side languages, databases) සංකීර්ණයි (සේවාදායක භාෂා, දත්ත සමුදායන්)
Interactivity අන්තර් ක්‍රියාකාරීත්වය	Limited සීමා සහිතයි	High (forms, user accounts) ඉහළයි (පෝරම, පරිශීලක ගිණුම්)
Performance කාර්ය සාධනය	Fast වේගවත්	Slower (server processing required) මන්දගාමීය (සේවාදායක සැකසුම් අවශ්‍යයි)
Maintenance නඩත්තු කිරීම	Manual updates අතින් යාවත්කාලීන කරයි	Backend interface or automatic updates පසුබිම් අතුරු මුහුණත හෝ ස්වයංක්‍රීය යාවත්කාලීන කිරීම්
Examples	Wikipedia, portfolios  WIKIPEDIA The Free Encyclopedia	E-commerce, social media, Streaming-services   

How PHP works

PHP ක්‍රියා කරන ආකාරය



Request Handling

ඉල්ලීම් හැසිරවීම

When a client (usually a web browser) requests a PHP page from a web server, the server identifies the file as a PHP script based on its .php extension.

සේවාලාභියෙකු (සාමාන්‍යයෙන් වෙබ් බ්‍රව්සරයක්) වෙබ් සේවාදායකයකින් PHP පිටුවක් ඉල්ලා සිටින විට, සේවාදායකය එහි .php දිගුව මත පදනම්ව ගොනුව PHP ස්ක්‍රිප්ට් එකක් ලෙස හඳුනා ගනී.

PHP Processing

PHP සැකසුම

The web server then sends the PHP file to the PHP interpreter, which is installed on the server. The PHP interpreter processes the PHP code within the file.

එවිට වෙබ් සේවාදායකය මගින් PHP ගොනුව සේවාදායකයේ ස්ථාපනය කර ඇති PHP පරිවර්තකය වෙත යවයි. PHP පරිවර්තකය ගොනුව තුළ ඇති PHP කේතය සකසයි.

Database Interaction

දත්ත සමුදා අන්තර්ක්‍රියා

If the PHP script includes database operations, PHP communicates with the database (Ex. MySQL) to retrieve or manipulate data.

PHP කේතයේ දත්ත සමුදා මෙහෙයුම් ඇතුළත් වේ නම්, PHP විසින් දත්ත ලබා ගැනීමට හෝ හැසිරවීමට දත්ත සමුදාය (උදා: MySQL) සමඟ සන්නිවේදනය කරයි.

Dynamic Content Generation

ගතික අන්තර්ගත උත්පාදනය

PHP generates dynamic content based on the script's logic and any data retrieved from databases or other sources.

PHP ස්ක්‍රිප්ට් වල තර්ක සහ දත්ත සමුදායන් හෝ වෙනත් මූලාශ්‍ර වලින් ලබා ගන්නා ඕනෑම දත්තයක් මත පදනම්ව ගතික අන්තර්ගතයන් ජනනය කරයි.

HTML Output

HTML ප්‍රතිදානය

The PHP interpreter converts the processed PHP code into an HTML output, which is the language understood by web browsers. PHP පරිවර්තකය විසින්, සැකසූ PHP කේතය HTML ප්‍රතිදානයක් බවට පරිවර්තනය කරයි, එය වෙබ් අතර්ක්ෂුව විසින් තේරුම් ගන්නා භාෂාව වේ.

Server Response

සේවාදායක ප්‍රතිචාරය

The web server receives the generated HTML output from the PHP interpreter and sends it back to the client's browser. වෙබ් සේවාදායකය PHP පරිවර්තකයෙන් ජනනය කරන ලද HTML ප්‍රතිදානය ලබාගෙන එය සේවාදායකයාගේ අතර්ක්ෂුව වෙත යවයි.

Client Display

සේවාලාභී සංදර්ශනය

The client's browser renders the received HTML output, displaying the final web page to the user. සේවාදායකයාගේ අතර්ක්ෂුව ලැබුණු නිමවූ ප්‍රතිදානය ලබා දෙන අතර අවසාන වෙබ් පිටුව පරිශීලකයාට පෙන්වයි.

Client-side processing and Server-side processing

සේවාලාභී අග්‍රයෙහි සැකසුම සහ සේවාදායක අග්‍රයෙහි සැකසුම

Client-side

HTML and CSS codes are executed directly on the client's computer, utilizing the browser to parse and render the content. HTML සහ CSS කේත අන්තර්ගතය විග්‍රහ කිරීමට සහ විදැහුම් කිරීමට අතර්ක්ෂුව භාවිතා කරමින් සෘජුවම සේවාලාභීයාගේ පරිගණකය මත ක්‍රියාත්මක වේ.

This process leverages the client's CPU and memory resources, causing the client's computer to consume power and generate heat as it processes and displays the webpage.

මෙම ක්‍රියාවලිය සේවාලාභීයාගේ සකසනය සහ මතක සම්පත් ප්‍රයෝජනයට ගන්නා අතර, වෙබ් පිටුව සැකසීමේදී සහ ප්‍රදර්ශනය කිරීමේදී සේවාලාභීයාගේ පරිගණකයෙහි බලය පරිභෝජනය වීමට සහ තාපය ජනනය වීමට හේතු වේ.

Server-side

PHP scripts are executed on the server before any content is sent to the client. සේවාදායකයා වෙත කිසියම් අන්තර්ගතයක් යැවීමට පෙර PHP කේත සේවාදායක පරිගණකය මත ක්‍රියාත්මක වේ.

This means the server's resources (its CPU, memory, and power) are used during the PHP execution, causing the server to consume power and generate heat.

මෙයින් අදහස් කරන්නේ සේවාදායකයේ සම්පත් (එහි සකසනය, මතකය සහ බලය) PHP ක්‍රියාත්මක කිරීමේදී භාවිතා වන අතර එය සේවාදායකයෙහි බලය පරිභෝජනය වීමට සහ තාපය ජනනය වීමට හේතු වේ.

Use cases of PHP

PHP හි භාවිත

Content Management Systems (CMS)

අන්තර්ගත කළමනාකරණ පද්ධති

Powering platforms like WordPress, Joomla, and Drupal for website and blog management.

වෙබ් අඩවි සහ බ්ලොග් කළමනාකරණය සඳහා WordPress, Joomla සහ Drupal වැනි වේදිකා බල ගැන්වීම.

E-commerce Platforms

ඊ-වාණිජ්‍යය වේදිකා

Used in Magento and WooCommerce for building online stores with product management and secure payment gateways.

නිෂ්පාදන කළමනාකරණය සහ ආරක්ෂිත ගෙවීම් ද්වාර සමඟ සබැඳි වෙළඳසැල් ගොඩනැගීම සඳහා Magento සහ WooCommerce හි භාවිත වේ.

Frameworks

Frameworks like Laravel, Symfony, and CodeIgniter streamline web application development with reusable components and database management.

Laravel, Symfony, සහ CodeIgniter වැනි නැවත භාවිත කළ හැකි සංරචක සහ දත්ත සමුදා කළමනාකරණය සමඟ වෙබ් යෙදුම් සංවර්ධනය විධිමත් කරයි.

Social Media Platforms

සමාජ මාධ්‍ය වේදිකා

Utilized by Facebook and parts of Twitter and LinkedIn for backend services and API responses.

පසුපෙළ සේවා සහ API ප්‍රතිචාර සඳහා Facebook සහ Twitter සහ LinkedIn කොටස් මගින් භාවිත කෙරේ.

Online Forums

Powering forum software like phpBB and vBulletin for community interactions.

ප්‍රජා අන්තර්ක්‍රියා සඳහා phpBB සහ vBulletin වැනි සංසද මාදුකාංග බල ගැන්වීම.

Web Services and APIs

වෙබ් සේවා සහ APIs

Developing RESTful APIs and web services that process requests and deliver dynamic content.

ඉල්ලුම් සකසන සහ ගතික අන්තර්ගතයන් ලබා දෙන RESTful API සහ වෙබ් සේවා සංවර්ධනය කිරීම.

Custom Web Applications

අභිරුචි වෙබ් යෙදුම්

Creating bespoke applications for specific business needs like CRMs and inventory management systems.

CRM සහ ඉන්වෙන්ටරි කළමනාකරණ පද්ධති වැනි විශේෂිත ව්‍යාපාරික අවශ්‍යතා සඳහා සුදුසු යෙදුම් නිර්මාණය කිරීම.

Setting up a virtual server

අතර්ගත සේවාදායකයක් සකසා ගැනීම

PHP runs on servers, which means we can't directly run PHP scripts on our local computer like we do with HTML or JavaScript.

PHP සේවාදායක මත ක්‍රියා කරයි, එනම් අපට HTML හෝ JavaScript මෙන් අපගේ ස්ථානීය පරිගණකයේ සෘජුවම PHP කේත ධාවනය කළ නොහැක.

To execute PHP code, we need a server environment that can process PHP scripts and handle HTTP requests.

PHP කේතය ක්‍රියාත්මක කිරීමට, අපට PHP ස්ක්‍රිප්ට් සැකසීමට සහ HTTP ඉල්ලීම් හැසිරවීමට හැකි සේවාදායක පරිසරයක් අවශ්‍ය වේ.

Since setting up an actual remote server can be costly and impractical for development and testing purposes, we temporarily turn our local computer into a virtual server.

සැබෑ දුරස්ථ සේවාදායකයක් පිහිටුවීම, සංවර්ධන සහ පරීක්ෂණ අරමුණු සඳහා මිල අධික සහ ප්‍රායෝගික නොවන බැවින්, අප අපගේ ස්ථානීය පරිගණකය තාවකාලිකව අතර්ගත සේවාදායකයක් බවට පත් කරනු ලැබේ.

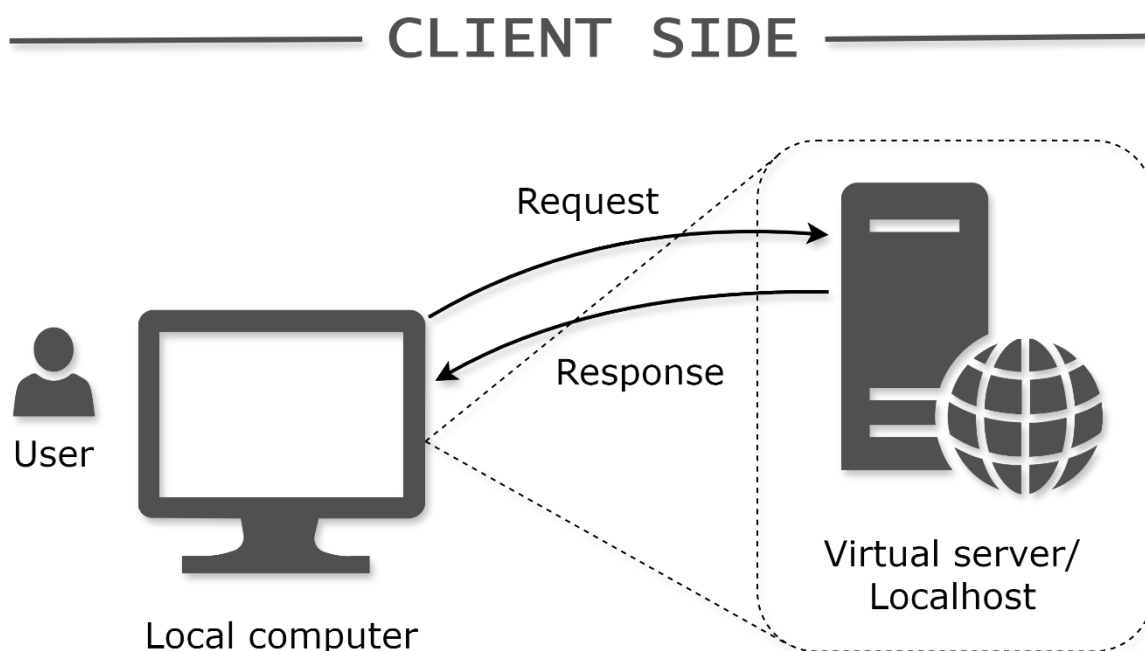
By creating a virtual host on our own computer, we simulate a web server environment, allowing us to send and receive HTTP requests using a web browser.

අපගේම පරිගණකයේ අතර්ගත සත්කාරකයක් නිර්මාණය කිරීමෙන්, වෙබ් බ්‍රවුසරයක් භාවිතයෙන් HTTP ඉල්ලීම් යැවීමට සහ ලැබීමට ඉඩ සලසමින් වෙබ් සේවාදායක පරිසරයක් අප පරිගණකය තුළ අනුකරණය කරයි.

This setup involves installing web server software, such as Apache or Nginx, along with PHP and a database server like MySQL. මෙම සැකසුම PHP සහ MySQL වැනි දත්ත සමුදා සේවාදායකයක් සමඟ Apache හෝ Nginx වැනි වෙබ් සේවාදායක මෘදුකාංග ස්ථාපනය කිරීම ඇතුළත් වේ.

Tools like XAMPP, WAMP, or MAMP make this process easier by bundling these components into a single package. Once installed, we can start the server software, which listens for incoming requests and processes PHP scripts.

XAMPP, WAMP, හෝ MAMP වැනි මෙවලම් මෙම සංරචක තනි පැකේජයකට එකතු කිරීමෙන් මෙම ක්‍රියාවලිය පහසු කරයි. ස්ථාපනය කළ පසු, ලැබෙන ඉල්ලීම්වලට සවන් දෙන සහ PHP ස්ක්‍රිප්ට් ක්‍රියාවට නංවන සේවාදායක මෘදුකාංගය ආරම්භ කළ හැක.



Server software packages සේවාදායක මෘදුකාංග පැකේජ

Tools like WAMP, XAMPP, MAMP, and LAMP are essential software packages that provide everything you need to create a virtual server on your local computer, enabling easy website development and application testing.

WAMP, XAMPP, MAMP, සහ LAMP වැනි මෙවලම් යනු ඔබේ ස්ථානීය පරිගණකයේ අතර්ගත සේවාදායකයක් නිර්මාණය කිරීමට අවශ්‍ය සියල්ල සපයනු ලබන අත්‍යවශ්‍ය මෘදුකාංග පැකේජ වන අතර, පහසු වෙබ් අඩවි සංවර්ධනය සහ යෙදුම් පරීක්ෂා කිරීම සක්‍රීය කරයි.

They create a virtual server on your local machine, mimicking the environment of a live web server.

ඒමගින් සජීවී වෙබ් සේවාදායකයක පරිසරය අනුකරණය කරමින් ඔබේ ස්ථානීය පරිගණකයේ අතර්ගත සේවාදායකයක් නිර්මාණය කරයි.

They bundle together a web server (Apache), a database server (MySQL or MariaDB), and server-side scripting languages (PHP, Perl, or Python).

ඒවා මගින් වෙබ් සේවාදායකයක් (Apache), දත්ත සමුදාය සේවාදායකයක් (MySQL හෝ MariaDB) සහ සේවාදායක අග්‍රයෙහි ස්ක්‍රිප්ටින් භාෂා (PHP, Perl හෝ Python) එකට එකතු කරයි.

They simplify the setup process by providing pre-configured packages that are easy to install and manage, reducing the complexity of setting up each component individually.

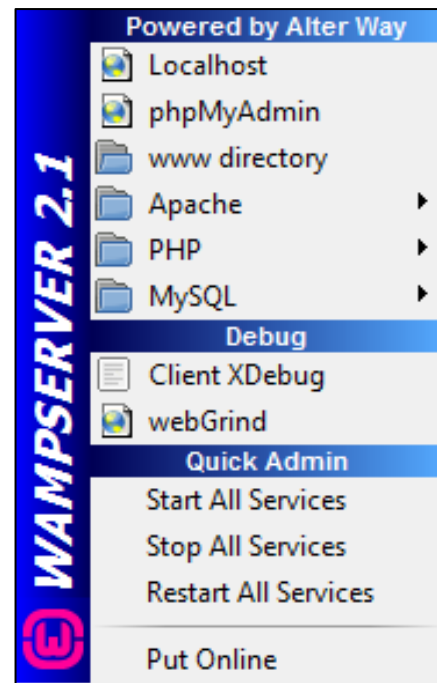
එක් එක් සංරචක තනි තනිව සැකසීමේ සංකීර්ණත්වය අඩු කරමින්, ස්ථාපනය කිරීමට සහ කළමනාකරණය කිරීමට පහසු වන පෙර-විනිසාසගත පැකේජ ලබා දීමෙන් මේවා මගින් සැකසුම් ක්‍රියාවලිය සරල කරයි.

WAMP (Windows, Apache, MySQL, PHP)

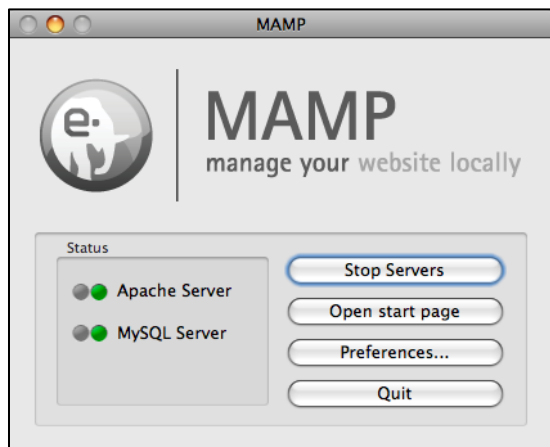


WAMP is tailored for Windows systems, combining Apache (web server), MySQL (database), and PHP (server-side scripting) to create a local server environment for developing PHP-based web applications on Windows.

WAMP Windows පද්ධති සඳහා සකස් කර ඇත, Apache (වෙබ් සේවාදායකය), MySQL (දත්ත සමුදාය) සහ PHP (සේවාදායක අග්‍රයෙහි ස්ක්‍රිප්ටින්) ඒකාබද්ධ කරමින් Windows මත ජ්‍යෙෂ්ඨ මත පදනම් වූ වෙබ් යෙදුම් සංවර්ධනය කිරීම සඳහා ස්ථානීය සේවාදායක පරිසරයක් නිර්මාණය කරයි.



MAMP (macOS, Apache, MySQL, PHP)



MAMP is designed for macOS, featuring Apache, MySQL (or MariaDB), and PHP to facilitate the creation of local server environments on Apple computers, ideal for developing and testing PHP applications.

MAMP නිර්මාණය කර ඇත්තේ MacOS සඳහා වන අතර, Apache, MySQL (හෝ MariaDB) සහ PHP විශේෂාංගීව Apple පරිගණකවල ස්ථානීය සේවාදායක පරිසරයන් නිර්මාණය කිරීමට පහසුකම් සැලසීමට, PHP යෙදුම් සංවර්ධනය කිරීම සහ පරීක්ෂා කිරීම සඳහා වඩාත් සුදුසුය.

LAMP (Linux, Apache, MySQL, PHP)

LAMP is widely used on Linux systems, comprising Apache, MySQL (or MariaDB), and PHP.

Apache, MySQL (හෝ MariaDB) සහ PHP වලින් සමන්විත Linux පද්ධති මත LAMP බහුලව භාවිතා වේ.



It serves as a robust platform for deploying and testing PHP applications locally or on Linux-based servers, known for its flexibility and stability.

එය නම්‍යශීලී බව සහ ස්ථායීතාවය සඳහා ප්‍රසිද්ධ PHP යෙදුම් ස්ථානීය හෝ Linux මත පදනම් වූ සේවාදායක මත යෙදවීමට සහ පරීක්ෂා කිරීමට ශක්තිමත් වේදිකාවක් ලෙස සේවය කරයි.

XAMPP (Cross-Platform, Apache, MySQL, PHP, Perl)



XAMPP is cross-platform, supporting Windows, macOS, and Linux. It includes Apache, MySQL (or MariaDB), PHP, and optionally Perl, providing developers with a unified stack for local web server setups across different operating systems.

XAMPP යනු Windows, macOS සහ Linux සඳහා සහය දක්වන හරස් වේදිකාවක් වේ. එයට Apache, MySQL (හෝ MariaDB), PHP සහ විකල්ප වශයෙන් Perl ඇතුළත් වේ, විවිධ මෙහෙයුම් පද්ධති හරහා ස්ථානීය වෙබ් සේවාදායක සැකසුම් සඳහා සංවර්ධකයින්ට ඒකාබද්ධ සංරචක කට්ටලයක් සපයයි.

Setting up WAMP

Download the WAMP server software from <https://sourceforge.net/projects/wampserver/>.

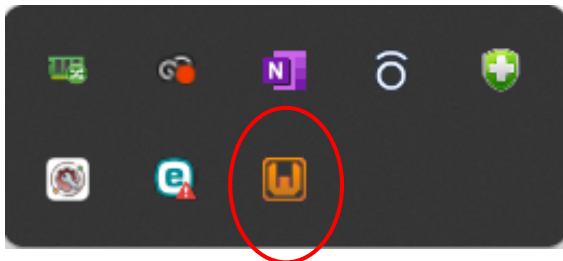
WAMP සේවාදායක මෘදුකාංගය

<https://sourceforge.net/projects/wampserver/>
වෙතින් බාගත කරගන්න.



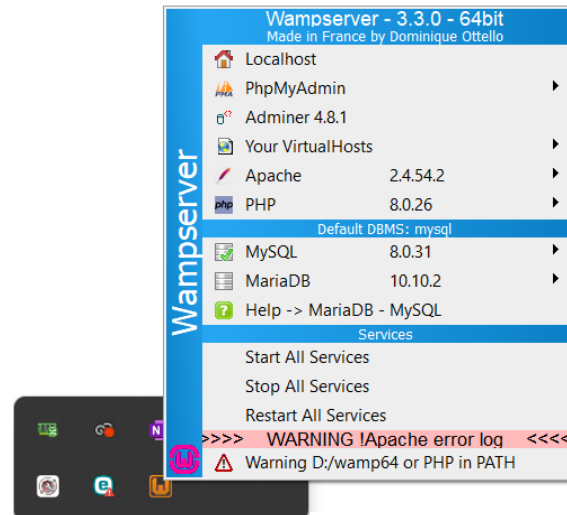
After installation, you should find a WAMP shortcut on your desktop or in your Start menu.

ස්ථාපනය කිරීමෙන් පසු, ඔබේ desktop එකේ හෝ ආරම්භක මෙනුවේ WAMP සඳහා කෙටිමගක් සොයා ගත හැකි විය යුතුය.



Click on the WAMP shortcut to launch the application.

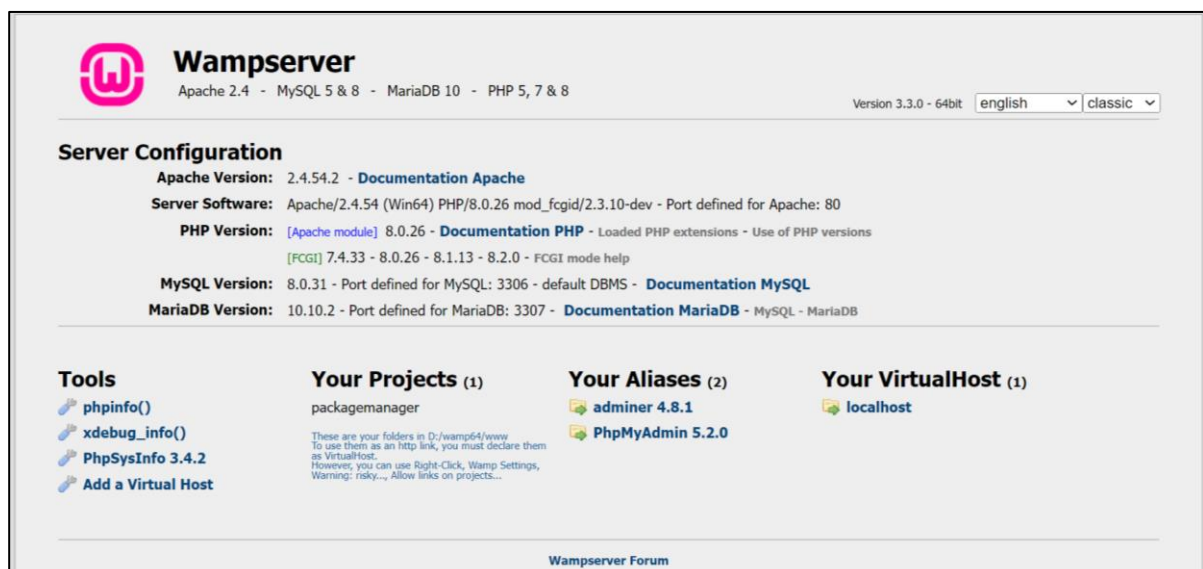
යෙදුම දියත් කිරීමට WAMP කෙටිමග මත ක්ලික් කරන්න.



Open your web browser and type `http://localhost/` into the address bar.

You should see the WAMP server homepage, indicating that the server is running correctly.

ඔබගේ වෙබ් බ්‍රව්සරය විවෘත කර ලිපිනය තීරුවේ `http://localhost/` ලෙස ටයිප් කරන්න. එවිට සේවාදායකය නිවැරදිව ක්‍රියාත්මක වන බව පෙන්නුම් කරමින් WAMP සේවාදායකයේ මුල් පිටුව දැකිය හැකි වේ.



Once your set-up is done, you can start writing PHP programs

ඔබගේ සැකසුම අවසන් වූ පසු, ඔබට PHP වැඩසටහන් ලිවීම ආරම්භ කළ හැක

By default, WAMP installs in C:\wamp64\ on 64-bit systems or C:\wamp\ on 32-bit systems.

පෙරහිමියෙන්, WAMP මෘදුකාංගය, 64-bit පද්ධති මත C:\wamp64\ හෝ 32-bit පද්ධති මත C:\wamp\ යන ස්ථානයන්හි ස්ථාපනය වේ.

Inside this directory, you'll find a folder named **www**. This is your virtual server for web projects.

මෙම නාමාවලිය තුළ, ඔබට **www** නමින් ගොනු නාමාවලියක් හමුවනු ඇත. මෙය වෙබ් ව්‍යාපෘති සඳහා ඔබගේ අතත්‍ය සේවාදායකයයි.

Inside the www folder, create a new directory for your PHP project.

www ෆෝල්ඩරය තුළ, ඔබේ PHP ව්‍යාපෘතිය සඳහා නව නාමාවලියක් සාදන්න.

For example, create a folder named myFirstProject.

උදාහරණයක් ලෙස, myFirstProject නමින් නාමාවලියක් සාදන්න.

The path should be something like

C:\wamp64\www\myproject.

එවිට ගොනු මාර්ගය

C:\wamp64\www\myproject ලෙස වේ.

Open a text editor (such as, Visual Studio Code, Sublime text or any IDE of your choice).

පෙළ සංස්කාරකයක් විවෘත කරන්න (such as, Visual Studio Code, Sublime text වැනි ඔබ කැමති ඕනෑම IDE එකක්).



Write your PHP code in the editor.

සංස්කාරකයේ ඔබේ PHP කේතය ලියන්න.

For example, create a file named index.php with the following content

උදාහරණයක් ලෙස, පහත අන්තර්ගතය සමඟ index.php නමින් ගොනුවක් සාදන්න

```
<?php
echo "Hello, World!";
?>
```

Save this file in the myFirstProject folder you created earlier.

මෙම ගොනුව ඔබ කලින් නිර්මාණය කළ myFirstProject නාමාවලියේ සුරකින්න.

(C:\wamp64\www\myproject\index.php)

Ensure WAMP is running, and the icon in the system tray is green, indicating all services are running.

WAMP ක්‍රියාත්මක වන බව සහ සියලු සේවාවන් ක්‍රියාත්මක වන බව පෙන්වුම් කරමින් එහි අයිකනය කොළ පාට බව සහතික කර ගන්න.

Open your web browser and type

http://localhost/myproject/index.php in the address bar.

ඔබගේ වෙබ් බ්‍රව්සරය විවෘත කර ලිපිත තීරුවේ http://localhost/myproject/index.php යන්න ටයිප් කරන්න.

You should see the output of your PHP code: Hello, World!.

දැන් ඔබේ PHP කේතයේ ප්‍රතිදානය Hello" World! ලෙසින් දිස් වනු ඇත.

NOTE: A file with a .php extension can contain HTML, CSS, and PHP code.

.php දිගුවක් සහිත ගොනුවක HTML, CSS සහ PHP කේත අඩංගු විය හැක.

Setting up XAMPP

Download the appropriate version for your operating system (Windows, macOS, Linux) ඔබගේ මෙහෙයුම් පද්ධතිය සඳහා සුදුසු අනුවාදය බාගත කරගන්න (Windows, macOS, Linux)
<https://www.apachefriends.org/download.html>

Run the downloaded installer.
බාගත කළ ස්ථාපකය ධාවනය කරන්න.

Follow the installation instructions. Choose the components you need (usually Apache, MySQL, and PHP).
ස්ථාපන උපදෙස් අනුගමනය කරන්න. ඔබට අවශ්‍ය සංරචක තෝරන්න (සාමාන්‍යයෙන් Apache, MySQL, සහ PHP).

Complete the installation.
ස්ථාපනය සම්පූර්ණ කරන්න.

Launch the XAMPP Control Panel from the start menu or installation directory.
ආරම්භක මෙනුවෙන් හෝ ස්ථාපන නාමාවලියෙන් XAMPP පාලන පැනලය දියත් කරන්න.

In the control panel, click the "Start" buttons next to Apache and MySQL modules.
පාලක පැනලයේ, Apache සහ MySQL මොඩියුල අසල ඇති "Start" බොත්තම් ක්ලික් කරන්න.

Navigate to the **htdocs** directory inside your XAMPP installation folder (usually C:\xampp\htdocs on Windows).

ඔබගේ XAMPP ස්ථාපන ෆෝල්ඩරය තුළ ඇති **htdocs** නාමාවලිය වෙතට ප්‍රවේශ වෙන්න.
(සාමාන්‍යයෙන් Windows වල C:\xampp\htdocs)

Create a new folder for your project (Ex. my_project).
ඔබගේ ව්‍යාපෘතිය සඳහා නව නාමාවලියක් සාදන්න (උදා. (පඨිවරදවැස්ව්)).

Inside your project folder, create a new PHP file (Ex. index.php).
ඔබගේ ව්‍යාපෘති නාමාවලිය තුළ, නව (ඡූෂ) ගොනුවක් සාදන්න (උදා. (සබැං.වයව)).

Type the following code

```
<?php  
echo "Hello, world!";  
?>
```

Launch your preferred web browser and in the address bar, type
http://localhost/my_project/index.php and press Enter.
ඔබ කැමති වෙබ් බ්‍රව්සරය දියත් කර ලිපිත තීරුවේ,
http://localhost/my_project/index.php යන්න ටයිප් කර Enter ඔබන්න.

You should see the output displayed in your browser.
ඔබගේ බ්‍රව්සරයේ දර්ශනය වෙන "Hello, world!" නම් ප්‍රතිදානය දැකිය යුතුය.



PHP syntax

PHP code should be written in-between `<?php` and `?>` tags

PHP කේත `<?php` සහ `?>` යන උසුලන අතර ලිවිය යුතුය

```
<?php
// PHP code goes here
?>
```

PHP code is embedded within HTML using PHP tags.

PHP කේත PHP උසුලන භාවිතයෙන් HTML තුළ අන්තර්ගත කර ඇත.

```
<!DOCTYPE html>
<html>
<head></head>
<body>
```

```
<?php
    echo "Hello World!";
?>
```

```
</body>
</html>
```

Semicolon (;)

In PHP, a semicolon is used to notify the computer that one instruction has finished and a new instruction is about to begin.

PHP හිදී, එක් උපදෙසක් අවසන් වූ පසු ඒ බව පරිගණකයට දැනුම් දීම සඳහාත් නව උපදෙසක් ආරම්භ වීමට යන බව දැනුම් දීම සඳහාත් semicolon එකක් භාවිතා වේ.

echo and print in PHP

In PHP, both echo and print are used to output data to the browser or command line.

PHP හිදී, ඔවුසරයට හෝ විධාන පේළියට දත්ත ප්‍රතිදානය කිරීමට echo සහ print යන දෙකම භාවිතා වේ.

echo

Type:

Language construct

Language construct එකකි

Usage:

Can take multiple parameters

පරාමිති කිහිපයක් ලබා ගත හැක

Return Value:

No return value (always returns void)

ආපසු අගයක් නැත (සෑම විටම විය හිස් වේ)

Performance:

Slightly faster than print

print ට වඩා තරමක් වේගවත්

print

Type:

Language construct

Language construct එකකි

Usage:

Can only take one argument

ගත හැක්කේ එක් පරාමිතියක් පමණි

Return Value:

Returns 1 (can be used in expressions)

1 ආපසු ලබාදේ (ප්‍රකාශනවල භාවිතා කළ හැක)

Performance:

Slightly slower than echo due to the return value

ආපසු අගය හේතුවෙන් echo ට වඩා තරමක් මන්දගාමී වේ

PHP variables

Variables in PHP are used to store data that can be manipulated and retrieved later in the script.

PHP හි විචල්‍යයන් භාවිතා කරනුයේ කේතයෙහි හැසිරවිය හැකි සහ පසුව ලබා ගත හැකි දත්ත ගබඩා කිරීමටය.

NOTE: Variables in PHP should be declared using a dollar sign (\$), informing the server computer that a variable is going to be created.

PHP හි විචල්‍යයන් ඩොලර් ලකුණකින් (\$) ආරම්භ විය යුතු අතර එමගින් විචල්‍යයක් සෑදීමට යන වග සේවාදායක පරිගණකය දැනුවත් කරයි.

PHP variables can hold values of any data type and do not require explicit declaration of the type.

PHP විචල්‍යයන්ට ඕනෑම දත්ත වර්ගයක අගයන් තබාගත හැකි අතර එම දත්ත වර්ගය ප්‍රථමයෙන් අර්ථ දැක්වීම අවශ්‍ය නොවේ.

Variable names in PHP are case-sensitive, meaning \$variable and \$Variable are treated as different variables.

PHP හි විචල්‍ය නාම අක්ෂර-සංවේදී වේ, එනම් \$variable සහ \$Variable විකිනෙකට වෙනස් විචල්‍යයන් ලෙස සලකනු ලැබේ.

But the PHP language itself is not case-sensitive

නමුත් PHP භාෂාව අක්ෂර-සංවේදී නොවේ

Ex. Echo, echo, eCHo are the same in PHP.

PHP හිදී Echo, echo, eCHo යනු එකම වේ.

Variables have a scope that defines where they can be accessed within the code.

(Scopes will be discussed later)

විචල්‍යයන්ට කේතය තුළ ප්‍රවේශ විය හැකි ස්ථාන නිර්වචනය කරන පරාසයක් ඇත.

(පරිපිළිබදව පසුව සාකච්ඡා කරනු ඇත)

Variable naming rules

විචල්‍ය නාමකරණ නීති

Must start with a letter or an underscore _
අකුරකින් හෝ යටි ඉරකින්_ ආරම්භ කළ යුතුය

Cannot start with a number
සංඛ්‍යාවකින් ආරම්භ කළ නොහැක

Can only contain alphanumeric characters and underscores (A-z, 0-9, and _)
අඩංගු විය හැක්කේ අක්ෂරාංක සලකුණු සහ යටි ඉර (A-z, 0-9, සහ _) පමණි

Reserved words of the language cannot be used as variable names.

භාෂාවේ ඇති ඇවිරුණු පද විචල්‍ය නම් ලෙස භාවිතා කළ නොහැක.

Ex.

```
<?php
/* Area and perimeter of a
Rectangle */

$length = 12;
$width = 5;
$a = $length * $width;
$p = 2*($length + $width);
echo "Area is $a";
echo "<br>";
echo "Perimeter is $p";
echo "<br>"; echo "<br>";
print "Area is $a";
print "<br>";
print "Perimeter is $p";
?>
```

Area is 60

Perimeter is 34

Area is 60

Perimeter is 34

PHP comments

PHP supports several types of comments
විවිධ වර්ගයේ විවරණ සඳහා සහාය වෙයි.

Single-line comments

- **Double Slash (//)**

Everything on the right of // will be treated as a comment.

// ට දකුණින් ඇති සියල්ල විවරණයක් ලෙස සලකනු ලබයි.

```
<?php
// This is a single-line comment
echo "Hello, World!";
?>
```

- **Hash (#)**

Similar to //, everything on the right of # will be treated as a comment.

// හා සමානව # ට දකුණින් ඇති සියල්ල විවරණයක් ලෙස සලකනු ලබයි.

```
<?php
# This is a single-line comment
echo "Hello, World!";
?>
```

Multi-line comments

- **Slash-Asterisk (/ * ... */)**

Encloses a block of text, allowing for multi-line comments.

පේළි කිහිපයක විවරණ සඳහා ඉඩ දෙමින් පාඨය සංවෘත කරයි.

```
<?php
/*
This is a multi-line comment
that spans multiple lines.
*/
echo "Hello, World!";
?>
```

NOTE: Multi-line comments can be used to comment a part of a single-line comment.
බහු-පේළි විවරණ තනි පේළියේ විවරණයක කොටසක් විවරණයක් බවට පත් කිරීමට භාවිතා කළ හැකිය.

```
<?php
```

```
$x = 10 + /* 50 */ + 20;
```

```
echo $x; # 30
```

```
?>
```

Here, /* 50 */ is a comment. Therefore, only 10 and 20 will be added.

මෙහි /* 50 */ යනු විවරණයකි. එබැවින් 10 සහ 20 පමණක් එකතු වේ.

Why use comments?

විවරණ භාවිතා කරන්නේ ඇයි?

- **Readability:** Improve code understanding.
කියවීමේ හැකියාව: කේත අවබෝධය වැඩි දියුණු කරයි.
- **Debugging:** Isolate and identify issues.
ගැටළු හඳුනාගැනීම: සමහර හඳුනා ගනියි.
- **Maintenance:** Facilitate future updates.
නඩත්තු කිරීම: අනාගත යාවත්කාලීන සඳහා පහසුකම් සැලසීම.
- **Collaboration:** Enhance team communication.
සහයෝගීතාව: කණ්ඩායම් සන්නිවේදනය වැඩි දියුණු කරයි.
- **Documentation:** Provide code context and purpose.
කේත සන්දර්භය සහ අරමුණ සපයීම.

PHP operators

PHP මෙහෙයවන

1. Arithmetic Operators

අංක ගණිතමය මෙහෙයවන

These operators are used to perform common arithmetic operations

මෙම මෙහෙයවන පොදු අංක ගණිත ක්‍රියා සිදු කිරීමට භාවිතා කරයි

Addition (+)

Adds two numbers.

සංඛ්‍යා දෙකක් එකතු කරයි.

```
<?php
echo 10 + 5; # 15
?>
```

Subtraction (-)

Subtracts one number from another.

එක් සංඛ්‍යාවකින් තවත් සංඛ්‍යාවක් අඩු කරයි.

```
<?php
echo 10 - 5; # 5
?>
```

Multiplication (*)

Multiply two numbers.

සංඛ්‍යා දෙකක් වැඩි කරයි.

```
<?php
echo 10 * 5; # 50
?>
```

Division (/)

Divides one number by another.

සංඛ්‍යාවක් තවත් සංඛ්‍යාවකින් බෙදනු ලබයි.

```
<?php
echo 10 / 5; # 2
?>
```

Modulus (%):

Returns the remainder of a division.

බෙදීමක ඉතිරි අගය ලබා දෙයි.

```
<?php
echo 11 % 5; # 1
?>
```

Exponentiation (**):

Raises one number to the power of another.

සංඛ්‍යාවක් තවත් සංඛ්‍යාවක බලය ලෙස ලබා දෙයි.

```
<?php
$a = 5;
$b = 2;
echo $a ** $b; # 25
?>
```

NOTE: In PHP, python's integer division (//) is not used as an operator. Instead, it is used to insert single-line comments.

PHP හිදී, පයිතන් වල නිඛිල බෙදීම (//) මෙහෙයවනයක් ලෙස භාවිතා නොවේ. ඒ වෙනුවට එය තනි පේළියේ විවරණ ඇතුළත් කිරීමට භාවිතා කරයි.

2. Assignment Operators

ආදේශක මෙහෙයවන

These operators are used to assign values to variables

මෙම මෙහෙයවන විචල්‍යයන් සඳහා අගයන් පැවරීමට භාවිතා කරයි

Basic Assignment (=)

Assigns the value on the right to the variable on the left.

වම් විචල්‍යයෙහි දැන් අගයෙහි දකුණු අගය වැඩි කර වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 10;
$b = $a;
?>
```

Addition Assignment (+=)

Adds the right operand to the left operand and assigns the result to the left operand.

වම් විචල්‍යයෙහි දැන් අගය දකුණු අගයට එකතු කර වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 10;
$a += 5;
echo $a; # 15
?>
```

Subtraction Assignment (-=)

Subtracts the right operand from the left operand and assigns the result to the left operand.

වම් විචල්‍යයෙහි දැන් අගයෙහි දකුණු අගය අඩු කර වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 10;
$a -= 5;
echo $a; # 5
?>
```

Multiplication Assignment (*=)

Multiplies the right operand by the left operand and assigns the result to the left operand.

වම් විචල්‍යයෙහි දැන් අගයෙහි දකුණු අගය වැඩි කර වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 10;
$a *= 5;
echo $a; # 50
?>
```

Division Assignment (/=)

Divides the left operand by the right operand and assigns the result to the left operand.

වම් විචල්‍යයෙහි දැන් අගය දකුණු අගයෙන් බෙදා වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 10;
$a /= 5;
echo $a; # 2
?>
```

Modulus Assignment (%=)

Calculates the modulus of the left operand and the right operand and assigns the result to the left operand.

වම් විචල්‍යයෙහි දැන් අගය දකුණු අගයෙන් බෙදා ඉතිරිය වම් විචල්‍ය වෙත ප්‍රතිඵලය පවරයි.

```
<?php
$a = 11;
$a %= 5;
echo $a; # 1
?>
```

3. Comparison Operators

සංසන්දනාත්මක මෙහෙයවන

These operators are used to compare two values

මෙම මෙහෙයවන අගයන් දෙකක් සංසන්දනය කිරීමට භාවිතා කරයි

Equal (==)

Checks if two values are equal.

අගයන් දෙකක් සමාන දැයි පරීක්ෂා කරයි

```
<?php
$a = 10;
$b = 10;
echo $a == $b; # 1
?>
```

Identical (===)

Checks if two values are equal and of the same type.

අගයන් දෙකක් සමාන සහ එකම වර්ගයේ දැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 10;
echo $a === $b; # 1
?>
```

Not Equal (!= or <>)

Checks if two values are not equal.

අගයන් දෙකක් සමාන නොවේදැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 5;
echo $a != $b; # 1
echo $a <> $b; # 1
?>
```

Not Identical (!==)

Checks if two values are not equal or not of the same type.

අගයන් දෙකක් සමාන නොවේද හැතහොත් එකම වර්ගයේ නොවේදැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 10.0;
echo $a !== $b; # 1
?>
```

Less Than (<)

Checks if the value on the left is less than the value on the right.

වම් පස ඇති අගය දකුණේ අගයට වඩා අඩු දැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 5;
echo $a < $b;
?>
```

Greater Than (>)

Checks if the value on the left is greater than the value on the right.

වම් පස ඇති අගය දකුණේ අගයට වඩා වැඩි දැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 5;
echo $a > $b; # 1
?>
```

Less Than or Equal To (<=)

Checks if the value on the left is less than or equal to the value on the right.

වම් පස ඇති අගය දකුණු පස ඇති අගයට වඩා අඩු හෝ සමානදැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 10;
echo $a <= $b; # 1
?>
```

Greater Than or Equal To (>=)

Checks if the value on the left is greater than or equal to the value on the right.

වම් පස ඇති අගය දකුණු පස ඇති අගයට වඩා වැඩි හෝ සමානදැයි පරීක්ෂා කරයි.

```
<?php
$a = 10;
$b = 11;
echo $a >= $b;
?>
```

4. Logical Operators

තාර්කික මෙහෙයවන

These operators are used to combine conditional statements

කොන්දේසි සහිත ප්‍රකාශ ඒකාබද්ධ කිරීමට මෙම මෙහෙයවන භාවිතා කරයි

And (&& or and)

Returns true if both operands are true.

මෙහෙයුම් දෙකම සත්‍ය නම් සත්‍ය ලෙස ලබා දේ.

```
<?php
$a = 10;
$b = 5;
$c = 4;
echo $a > $b && $b > $c; # 1
?>
```

Or (|| or or)

Returns true if at least one of the operands is true.

අවම වශයෙන් එක් මෙහෙයුමක්වත් සත්‍ය නම් සත්‍ය ලෙස ලබා දේ.

```
<?php
$a = 10;
$b = 5;
$c = 4;
echo $a > $b || $b < $c; # 1
?>
```

Not (!)

Reverses the logical state of its operand.

එහි operand හි තාර්කික තත්ත්වය ආපසු හරවයි.

```
<?php
$b = 0;
echo !$b; # 1
?>
```

Xor (xor)

Returns true if either operand is true, but not both.

ඔපෙරාන්ඩ් දෙකෙන් එකක් සත්‍ය නම් සත්‍ය ලබා දෙයි, නමුත් දෙකම නොවේ.

```
<?php
$a = 10;
$b = 5;
$c = 4;
echo $a > $b xor $b < $c; # 1
echo $a > $b xor $b > $c;
?>
```

5. Increment/Decrement Operators

These operators are used to increment or decrement a variable's value:

විචල්‍යයක අගය වැඩි කිරීමට හෝ අඩු කිරීමට මෙම මෙහෙයවන භාවිතා කරයි

Increment (++)

Increases a variable's value by one.

විචල්‍යයක අගය එකකින් වැඩි කරයි.

They are two types.

ආකාර දෙකකි

1. Pre-increment

In pre-increment, the variable's value is incremented first, and then the new value is used in the expression.

Pre-increment හිදී, විචල්‍යයේ අගය පළමුව වැඩි කරනු ලබන අතර, පසුව නව අගය ප්‍රකාශනයේ භාවිතා වේ.

```
<?php
$number = 5;
echo ++$number; // Outputs: 6
echo $number;   // Outputs: 6
?>
```

2. Post-increment

In post-increment, the current value of the variable is used in the expression first, and then the variable's value is incremented.

Post-increment හිදී, විචල්‍යයේ වත්මන් අගය ප්‍රකාශනයේ පළමුව භාවිතා කරනු ලබන අතර, පසුව විචල්‍යයේ අගය වැඩි වේ.

```
<?php
$number = 5;
echo $number++; // Outputs: 5
echo $number;   // Outputs: 6
?>
```

Decrement (--)

Decreases a variable's value by one.
විචලනයක අගය එකකින් අඩු කරයි.

They are two types.
ආකාර දෙකකි

1. Pre-decrement

In pre-decrement, the variable's value is decremented first, and then the new value is used in the expression.

Pre-decrement හිදී, විචලනයේ අගය පළමුව අඩු කරනු ලබන අතර, පසුව නව අගය ප්‍රකාශනයේ භාවිතා වේ.

```
<?php
    $number = 5;
    echo --$number; // Outputs: 4
    echo $number;    // Outputs: 4
?>
```

2. Post-decrement

In post-decrement, the current value of the variable is used in the expression first, and then the variable's value is decremented.

Post-decrement හිදී, විචලනයේ වත්මන් අගය ප්‍රකාශනයේ පළමුව භාවිතා කරනු ලබන අතර, පසුව විචලනයේ අගය අඩු වේ.

```
<?php
    $number = 5;
    echo $number--; // Outputs: 5
    echo $number;    // Outputs: 4
?>
```

6. String Operators

These operators are used to manipulate strings

මෙම මෙහෙයවන තන්තු හැසිරවීමට භාවිතා කරයි

Concatenation (.)

Concatenates two strings.
තන්තු දෙකක් එකට සම්බන්ධ කරයි.

```
<?php

$firstName = "John";
$lastName = "Hamilton";
$fullName = $firstName . $lastName;
echo "Full Name is $fullName";

?>
```

Full Name is John Hamilton

Concatenation Assignment (.=)

Appends the right string to the left string and assigns the result to the left variable.
දකුණෙහි තන්තුව වම් තන්තුවට එකතු කර වම් විචලනයට ප්‍රතිඵලය ආදේශ කරයි.

```
<?php

$fullName = "John";
$fullName .= " Hamilton";
echo "Full Name is $fullName";

?>
```

Full Name is John Hamilton

var_dump() in PHP

The `var_dump()` function in PHP is used to display structured information (type and value) about one or more variables.

PHP හි `var_dump()` ශ්‍රිතය විවිධ වීක්ෂණීය තොරතුරු (දත්ත වර්ගය සහ අගය) පෙන්වීමට භාවිතා කරයි.

It provides a detailed output, making it especially useful for debugging purposes.

එය සවිස්තරාත්මක ප්‍රතිදානයක් සපයයි, එය දෝෂහරණය කිරීමේ අරමුණු සඳහා විශේෂයෙන් ප්‍රයෝජනවත් වේ.

Syntax

```
<?php
var_dump($variable1, $variable2, ...);
?>
```

Ex.

```
<?php
$number = 42;
$string = "Hello, World!";
$float = 3.14;
$boolean = true;

var_dump($number);
var_dump($string);
var_dump($float);
var_dump($boolean);
?>
```

```
int(42)
string "Hello, world!" (length=13)
float 3.14
boolean true
```

Data types in PHP

PHP වල දත්ත වර්ග

String

A sequence of characters enclosed in single or double quotes.

තනි හෝ ද්විත්ව උද්ධෘතවලින් වට කර ඇති සලකුණු අනුපිළිවෙලකි.

```
<?php
$name = "John";
var_dump($name);
?>
```

```
string "John" (length=4)
```

Integer

A non-decimal number between

-2,147,483,648 and 2,147,483,647.

-2,147,483,648) සහ 2,147,483,647 අතර දශම නොවන සංඛ්‍යාවකි.

```
<?php
$number = 42;
var_dump($number);
?>
```

```
int 42
```

Float (Double)

A number with a decimal point.

දශම ස්ථානයක් සහිත සංඛ්‍යාවකි.

```
<?php
$floatNumber = 3.14;
var_dump($floatNumber);
?>
```

```
float 3.14
```

Boolean

Represents two possible states: TRUE or FALSE.

පැවතිය හැකි අවස්ථා දෙකක් නියෝජනය කරයි: සත්‍ය හෝ අසත්‍ය.

```
<?php
    $bool1 = true;
    $bool2 = false;
    Var_dump($bool1, $bool2);
?>
```

```
boolean true
boolean false
```

Array

A collection of values. Each value can be accessed using an index or key.

අගයන් කිහිපයක එකතුවකි. සෑම අගයකටම දර්ශකයක් හෝ key එකක් භාවිතයෙන් ප්‍රවේශ විය හැක.

There are three main types of arrays in php. php හි ප්‍රධාන ආරා වර්ග තුනක් ඇත.

1. Indexed arrays
2. Multi-dimensional arrays
3. Associative arrays

1. Indexed arrays

These are arrays with numeric indices.

මේවා සංඛ්‍යාංකමය දර්ශක සහිත ආරා වන්නේ වේ.

```
<?php

$indexedArray = array(10, "apple",
3.14, true);

var_dump($indexedArray);

?>
```

```
array (size=4)
  0 => int 10
  1 => string 'apple' (length=5)
  2 => float 3.14
  3 => boolean true
```

2. Multi-dimensional arrays

These are Arrays containing one or more arrays.

මේවා ආරවන් එකක් හෝ කිහිපයක් අඩංගු Arrays වේ.

```
<?php
$multiArray = array(
    "nums" => array(10, 20, 30),
    "letters" => array("a", "b", "c")
);
Var_dump($multiArray);
?>
```

```
array (size=2)
  'nums' =>
    array (size=3)
      0 => int 10
      1 => int 20
      2 => int 30
  'letters' =>
    array (size=3)
      0 => string 'a' (length=1)
      1 => string 'b' (length=1)
      2 => string 'c' (length=1)2.
```

3. Associative arrays

Arrays that use named keys instead of indices.

දර්ශක වෙනුවට keys භාවිතා කරන ආරවන් වේ.

```
<?php
$assocArray = array("a" => 10,
    "b" => "apple", "c" => 3.14,
    "d" => true);
Var_dump($assocArray);
?>
```

```
array (size=4)
  'a' => int 10
  'b' => string 'apple' (length=5)
  'c' => float 3.14
  'd' => boolean true
```

Data types summary

Data Type	Description	Example
String	Sequence of characters	<code>\$string = "Hello, World!";</code>
Integer	Non-decimal number	<code>\$integer = 42;</code>
Float	Number with a decimal point	<code>\$float = 3.14;</code>
Boolean	TRUE or FALSE	<code>\$boolean = true;</code>
Array	Collection of values	<code>\$array = array("apple", "banana", "cherry");</code>

if-else conditions

If conditions allow you to execute specific blocks of code based on whether a given condition evaluates to true or false.

දී ඇති කොන්දේසියක් සත්‍ය හෝ අසත්‍ය ලෙස ඇගයීමට ලක් කරන්නේද යන්න මත පදනම්ව නිශ්චිත කේත කොටස් ක්‍රියාත්මක කිරීමට ඔබට ඉඩ ලබා දෙයි.

This control structure is fundamental for creating dynamic and responsive applications.

ගතික සහ ප්‍රතිචාරාත්මක යෙදුම් නිර්මාණය කිරීම සඳහා මෙම පාලන ව්‍යුහය මූලික වේ.

Syntax

```
<?php
if (condition) {
    /* code to be executed if
    condition is true */
} else {
    /* code to be executed if
    condition is false */
}
?>
```

Ex.

```
<?php

$isRaining = true;

if ($isRaining) {
    echo "Bring an umbrella";
} else {
    echo "Bring a cap";
}
?>
```

Bring an umbrella

if-elseif-else Statement

You can test multiple conditions using elseif.

elseif භාවිතයෙන් ඔබට කොන්දේසි කිහිපයක් පරීක්ෂා කළ හැක.

Ex.

```
<?php
$score = 85;

if ($score >= 90) {
    echo "Grade: A";
} elseif ($score >= 80) {
    echo "Grade: B";
} elseif ($score >= 70) {
    echo "Grade: C";
} else {
    echo "Grade: F";
}
?>
Grade: B
```

Nested If Statements

If statements inside another if statement are known as nested if statements.

වෙනත් If ප්‍රකාශනයක් තුළ ඇති if ප්‍රකාශනයන් nested if ප්‍රකාශ ලෙස හැඳින්වේ.

```
<?php
$age = 20;
$hasID = true;

if ($age >= 18) {
    if ($hasID) {
        echo "You can enter.";
    } else {
        echo "You need an ID to
        enter.";
    }
} else {
    echo "You are not old enough
    to enter.";
}
?>
```

Shorthand If

The shorthand if statement in PHP is known as the **ternary operator**.

PHP හි ඇති කෙටි if ප්‍රකාශය ලෙස හැඳින්වේ.

It allows you to write a compact form of the if-else statement in a single line.

එය ඔබට if-else ප්‍රකාශයේ සංයුක්ත ආකෘතියක් තනි පේළියකින් ලිවීමට ඉඩ සලසයි.

Syntax

```
<?php
condition ? true_value : false_value;

?>
```

condition: The expression to be evaluated.

ඇගයීමට ලක් කළ යුතු ප්‍රකාශනය.

true_value: The value to return if the condition is true.

කොන්දේසිය සත්‍ය නම් ආපසු ලබා දිය යුතු අගය.

false_value: The value to return if the condition is false.

කොන්දේසිය අසත්‍ය නම් ආපසු ලබා දිය යුතු අගය.

```
<?php

$age = 20;

$message = ($age >= 18) ? "You are
an adult." : "You are a minor.";

echo $message;

?>
```

You are an adult.

Nested Ternary Operators

You can nest ternary operators, but this can reduce readability.

Ternary operators, nest කළ හැකිය, නමුත් මෙය කියවීමේ හැකියාව අඩු කළ හැක.

```
<?php
$score = 85;

$grade = ($score >= 90) ? "A" :
(($score >= 80) ? "B" :
(($score >= 70) ? "C" : "F"));

echo $grade;

?>
```

B

With echo Statement

```
<?php

$loggedIn = true;

echo $loggedIn ? "Welcome back!" :
"Please log in.";

?>
```

Welcome back!

Switch Statement

The switch statement in PHP is used to perform different actions based on different conditions.

PHP හි Switch ප්‍රකාශය විවිධ කොන්දේසි මත පදනම්ව විවිධ ක්‍රියා සිදු කිරීමට භාවිතා කරයි.

It is a cleaner and more readable way to compare a variable against multiple values than using multiple if statements.

එය if ප්‍රකාශන කිහිපයක් භාවිතා කරනවාට වඩා බහු අගයන් සමඟ විචල්‍යයක් සංසන්දනය කිරීමට වඩාත් හොඳ සහ කියවිය හැකි ක්‍රමයකි.

Syntax

```
<?php
switch (variable) {
    case value1:
        /* code to be executed if
        variable equals value1 */
        break;
    case value2:
        /* code to be executed if
        variable equals value2 */
        break;
    default:
        /* code to be executed if
        variable does not match any
        case */
}
?>
```

How It Works

එය ක්‍රියා කරන ආකාරය

switch: The keyword that initiates the switch statement.
switch ප්‍රකාශය ආරම්භ කරන මූල පදය.

variable: The variable whose value you want to compare against different cases.
විවිධ අවස්ථා සමඟ සංසන්දනය කිරීමට අවශ්‍ය විචල්‍යය.

case: Each possible value that the variable can have. If the variable matches a case value, the corresponding block of code is executed.
විචල්‍යයට තිබිය හැකි සෑම අගයක්ම. විචල්‍යය case හි අගයකට ගැළපේ නම්, අදාළ කේත කොටස ක්‍රියාත්මක වේ.

break: This keyword terminates the case once it is executed. Without break, PHP will continue to execute the following cases, a behavior known as "fall-through."
මෙම මූල පදය එය ක්‍රියාත්මක කළ පසු case එක අවසන් කරයි. break ප්‍රකාශයෙන් තොරව, PHP පහත සඳහන් අවස්ථා ක්‍රියාත්මක කිරීම දිගටම කරගෙන යනු ඇත, එය "fall-through" ලෙස හැඳින්වේ.

default: An optional case that will be executed if the variable does not match any of the specified cases.
විචල්‍යය නිශ්චිත අවස්ථා කිසිවක් සමඟ නොගැළපේ නම් ක්‍රියාත්මක වන විකල්ප අවස්ථාවකි.

Ex.

```
<?php
$day = "Wednesday";

switch ($day) {
    case "Monday":
        echo "Today is Monday.";
        break;
    case "Tuesday":
        echo "Today is Tuesday.";
        break;
    case "Wednesday":
        echo "Today is Wednesday.";
        break;
    case "Thursday":
        echo "Today is Thursday.";
        break;
    case "Friday":
        echo "Today is Friday.";
        break;
    default:
        echo "Invalid day.";
}
?>
```

Today is Wednesday.

Example with numbers.

```
<?php
$number = 2;

switch ($number) {
    case 1:
        echo "Number is 1";
        break;
    case 2:
        echo "Number is 2";
        break;
    case 3:
        echo "Number is 3";
        break;
    default:
        echo "Number is not 1, 2,
        or 3";
}
?>
```

Number is 2

Example with Default Case

```
<?php
$color = "purple";

switch ($color) {
    case "red":
        echo "Color is red.";
        break;
    case "blue":
        echo "Color is blue.";
        break;
    case "green":
        echo "Color is green.";
        break;
    default:
        echo "Color is not red,
        blue, or green.";
}
?>
```

Color is not red, blue, or green.

Loops in PHP

Loops are used to execute the same block of code repeatedly as long as a certain condition is met.

යම් කොන්දේසියක් සපුරා ඇති තාක් එකම කේතය නැවත නැවතත් ක්‍රියාත්මක කිරීමට ලූප භාවිතා වේ.

There are four main types of loops in PHP
PHP හි ප්‍රධාන ලූප වර්ග හතරක් ඇත

1. for
2. foreach
3. while
4. do-while

For loop

The for loop is used when you know in advance how many times you want to execute a statement or a block of statements.

ප්‍රකාශයක් හෝ ප්‍රකාශ සමූහයක් ක්‍රියාත්මක කිරීමට කොපමණ වාර ගණනක් අවශ්‍ය දැයි කලින් දන්නා විට for loop භාවිතා වේ.

Syntax

```
<?php
for (initialization; condition;
increment) {
    // code to be executed
}
?>
```

Ex1.

```
<?php
for ($x = 0; $x <= 10; $x++) {
    echo "The number is: $x <br>";
}
?>
```

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
The number is: 6
The number is: 7
The number is: 8
The number is: 9
The number is: 10

Ex2.

```
<?php
for ($x = 1; $x <= 10; $x++) {
    echo "5 x $x = " , (5*$x);
    echo "<br>";
}
?>
```

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50

While loop

The while loop executes a block of code as long as the specified condition is true.

ලබා දී ඇති කොන්දේසිය සත්‍ය වන තාක් while ලූපය මගින් කේත කොටසක් ක්‍රියාත්මක කරයි.

Syntax

```
<?php
while (condition) {
    // code to be executed
}
?>
```

We can control the number of times that a while loop executes in three ways.

while ලූපයක් ක්‍රියාත්මක වන වාර ගණන ආකාර තුනකින් පාලනය කළ හැකිය.

1. By changing the control variable.

පාලන විචල්‍යය වෙනස් කිරීමෙන්.

2. By changing the condition.

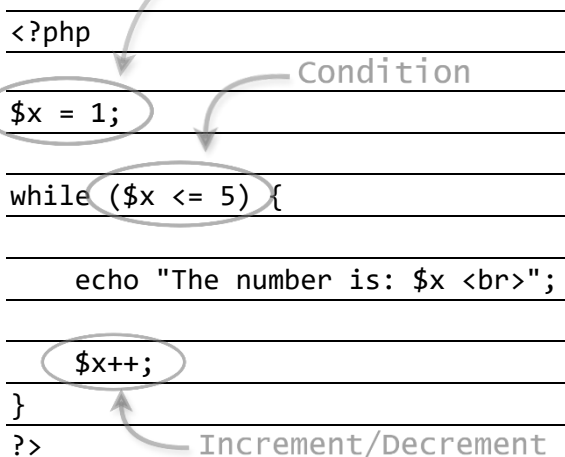
කොන්දේසිය වෙනස් කිරීමෙන්.

3. By changing the increment/decrement of the control variable.

පාලන විචල්‍යයේ වැඩිකිරීම්/අඩුකිරීම් වෙනස් කිරීමෙන්.

Ex1.

```
<?php
$x = 1;
while ($x <= 5) {
    echo "The number is: $x <br>";
    $x++;
}
?>
```



The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5

Ex2.

```
<?php
$x = 1;
while ($x <= 5) {
    echo "5 x $x = ", 5 * $x;
    echo "<br>";
    $x++;
}
?>
```

5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25

Do-While Loop

The do-while loop will always execute the block of code inside it once, then it will check the condition and repeat the loop as long as the specified condition is true.

do-while ලූපය සෑම විටම වී තුළ ඇති කේත කොටස එක් වරක් ක්‍රියාත්මක කරයි, පසුව එය කොන්දේසිය පරීක්ෂා කර නිශ්චිත කොන්දේසිය සත්‍ය වන තාක් ලූපය නැවත ක්‍රියාත්මක කරයි.

Syntax

```
<?php
do {
    // code to be executed
} while (condition);
?>
```

Ex.

```
<?php
$x = 1;
do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5

Let's consider a scenario where the condition is false at the very beginning. ආරම්භයේදීම කොන්දේසිය අසත්‍ය වන අවස්ථාවක් සලකා බලමු.

```
<?php
$weight = 52;
$maxWeight = 45;
do {
    echo "Weight($weight) is less
    than max weight($maxWeight).";
    echo "<br>";
    $weight++;
} while ($weight <= $maxWeight);

echo "Weight($weight) exceeded max
weight($maxWeight).";
?>
```

Weight(52) is less than max weight(45).
Weight(53) exceeded max weight(45).

In the above example, normal weight is already higher than the max weight when the program starts.

ඉහත උදාහරණයේ දී, වැඩසටහන ආරම්භ වන විටදීම බර උපරිම බරට වඩා වැඩි ය.

But the do-while loop was executed once, outputting that the weight is less than max weight.

නමුත් බර උපරිම බරට වඩා අඩු බව ප්‍රතිදානය කරමින් do-while ලූපයක් එක වරක් ක්‍රියාත්මක විය.

So, a do-while loop executes the instruction inside it at least one time.

එබැවින්, do-while ලූපයක් අවම වශයෙන් එක් වරක්වත් එහි ඇති උපදෙස් ක්‍රියාත්මක කරයි.

Foreach Loop

The foreach loop is used to loop through arrays.

foreach ලූපය ආරාවන් හරහා පුනරාවර්තනය කිරීමට භාවිතා කරයි.

For every iteration, the value of the current array element is assigned to a variable, and the array pointer is moved by one, until it reaches the last array element.

සෑම පුනරාවර්තනයක් සඳහාම, වත්මන් අරා මූලාවයවයේ අගය විචල්‍යයකට පවරනු ලබන අතර, අවසාන අරා මූලාවයවය වෙත ළඟා වන තෙක් අරා දර්ශකය එකකින් වැඩි කරනු ලැබේ.

Syntax

```
<?php
foreach ($array as $value) {
    // code to be executed
}
?>
```

Here, each element in the \$array variable is assigned to the \$value variable in each iteration until the loop reaches the last element.

මෙහිදී, \$array විචල්‍යයේ ඇති සෑම මූලාවයවයක්ම, ලූපය අවසාන මූලාවයවය වෙත ළඟා වන තුරු එක් එක් පුනරාවර්තනයේදී \$array විචල්‍යයට පවරනු ලැබේ.

Ex.

```
<?php
$colors = array("red", "green",
"blue", "yellow");

foreach ($colors as $color) {
    echo "$color <br>";
}
?>
```

```
red
green
blue
yellow
```

Associative Array Example:

```
<?php

$ages = array("Peter"=>"35",
"Ben"=>"37", "Joe"=>"43");

foreach ($ages as $name => $age){
    echo "$name is $age years
    old.<br>";
}

?>
```

```
Peter is 35 years old.
Ben is 37 years old.
Joe is 43 years old.
```


Nested Loops

You can nest loops inside each other.
ඔබට ලූප තුළ ලූප කඳෙලි කළ හැක.

This is useful when working with multidimensional arrays or when you need to perform complex iterations.

Multidimensional arrays සමඟ ක්‍රියා කරන විට හෝ සංකීර්ණ පුනරාවර්තන සිදු කිරීමට අවශ්‍ය විට මෙය ප්‍රයෝජනවත් වේ.

Ex.

```
<?php
for ($i = 1; $i <= 3; $i++) {
    echo "Outer Loop iteration
    $i<br>";

    for ($j = 1; $j <= 3; $j++) {
        echo "- Inner iteration
        $j<br>";
    }
    echo "<br>";
}

?>
```

Outer Loop iteration 1
- Inner iteration 1
- Inner iteration 2
- Inner iteration 3

Outer Loop iteration 2
- Inner iteration 1
- Inner iteration 2
- Inner iteration 3

Outer Loop iteration 3
- Inner iteration 1
- Inner iteration 2
- Inner iteration 3

Break and Continue

break

Used to exit the loop immediately.
ලූපයෙන් වහාම ඉවත් වීමට භාවිතා කරයි.

continue

Used to skip the current iteration and proceed to the next iteration of the loop.
වත්මන් පුනරාවර්තනය මඟ හැර ලූපයේ ඊළඟ පුනරාවර්තනය වෙත යාමට භාවිතා කරයි.

Ex. - break

```
<?php
for ($x = 0; $x < 10; $x++) {
    if ($x == 5) {
        break;
    }
    echo "The number is: $x <br>";
}

?>
```

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4

Ex. - continue

```
<?php
for ($x = 0; $x < 7; $x++) {
    if ($x == 5) {
        continue;
    }
    echo "The number is: $x <br>";
}

?>
```

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 6

PHP Functions

PHP ශ්‍රිත

Functions are blocks of code that perform specific tasks and can be called upon whenever needed.

ශ්‍රිත යනු නිශ්චිත කාර්යයන් ඉටු කරන කේත කොටස් වන අතර අවශ්‍ය විටෙක ඒවා ඇමතීමට හැක.

They help in organizing code, making it reusable, and improving readability and maintainability.

ඒවා කේතය සංවිධානය කිරීමට, නැවත භාවිතා කළ හැකි බවට පත් කිරීමට සහ කියවීමේ හැකියාව සහ හඬන්නා කිරීමේ හැකියාව වැඩි දියුණු කිරීමට උපකාරී වේ.

Defining and Calling Functions

ශ්‍රිත නිර්වචනය කිරීම සහ ඇමතීම

Basic Function Definition

To define a function in PHP, use the function keyword, followed by the function name, parentheses (), and a block of code within {}.

PHP හි ශ්‍රිතයක් නිර්වචනය කිරීම සඳහා, function මූල පදය භාවිතා කරන්න, ඉන්පසු ශ්‍රිත නාමය, වරහන් () සහ {} තුළ කේත කොටසක් භාවිතා කරන්න.

Syntax

```
<?php
function functionName() {
    // code to be executed
}
?>
```

Ex.

```
<?php
function greet() {
    echo "Hello, world!";
}

// Calling the function
greet();
?>
```

Hello, world!

Functions can be categorized into four types based on parameters and the return value.

පරාමිති සහ return අගය මත පදනම්ව ශ්‍රිත වර්ග හතරකට කාණ්ඩ කළ හැකිය.

1. Functions with Only Parameters

These functions accept input values but do not return a value.

මෙම ශ්‍රිතයන් ආදාන අගයන් පිළිගන්නා නමුත් අගයක් ආපසු ලබා නොදේ.

They perform operations based on the provided parameters.

සපයන ලද පරාමිතීන් මත පදනම්ව ඒවා මෙහෙයුම් සිදු කරයි.

Ex.

```
<?php
function greet($name) {
    echo "Hello, $name!";
}

greet("Alice");
?>
```

Hello, Alice!

2. Functions with Only Return Values

These functions do not accept any parameters but return a value after execution.

මෙම ශ්‍රිතයන් කිසිදු පරාමිතියක් නොපිළිගන්නා නමුත් ක්‍රියාත්මක කිරීමෙන් පසු අගයක් ආපසු ලබා දෙයි.

Ex.

```
<?php

function getGreeting() {
    return "Hello, world!";
}

echo getGreeting();

?>
```

Hello, world!

3. Functions with Parameters and Return Values

Functions with both parameters and return values accept input values, perform operations, and return a result.

පරාමිති සහ ආපසු අගයන් දෙකම සහිත ශ්‍රිත ආදාන අගයන් පිළිගනී, මෙහෙයුම් සිදු කරයි, සහ ප්‍රතිඵලයක් ආපසු ලබා දෙයි.

Ex.

```
<?php

function add($a, $b) {
    return $a + $b;
}

$result = add(3, 4);
echo "Result is $result.";

?>
```

Result is 7.

4. Functions without Parameters and Return Values

Functions without parameters and return values neither accept input values nor return a result.

පරාමිති සහ ආපසු අගයන් නොමැති ශ්‍රිත ආදාන අගයන් පිළිගන්නේවත් ප්‍රතිඵලයක් ආපසු ලබා දෙන්නේවත් නැත.

They perform predefined tasks.

ඒවා පෙර නිර්වචනය කරන ලද කාර්යයන් ඉටු කරයි.

Ex.

```
<?php

function greet() {
    echo "Hello, world!";
}

greet();

?>
```

Hello, world!

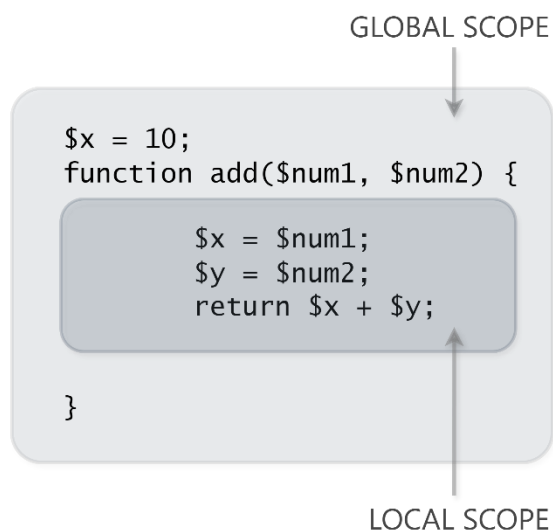
Variable Scope

Variables inside a function have local scope, meaning they can only be accessed within that function. They are known as **local variables**.

ශ්‍රිතයක් තුළ ඇති විචල්‍යයන්ට ස්ථානීය පරාසයක් ඇත, එනම් ඒවාට ප්‍රවේශ විය හැක්කේ එම ශ්‍රිතය තුළදී පමණි. ඒවා **ස්ථානීය විචල්‍යයන්** නම් වේ.

Variables outside of functions have global scope and cannot be accessed inside functions unless explicitly specified. They are called **global variables**.

ශ්‍රිතයෙන් පිටත විචල්‍යයන්ට ගෝලීය පරාසයක් ඇති අතර පැහැදිලිව සඳහන් කර ඇත්නම් මිස ශ්‍රිත තුළට ප්‍රවේශ විය නොහැක. ඒවා **ගෝලීය විචල්‍යයන්** නම් වේ.



`$x` in the global scope and `$x` in the local scope are two different variables.

ගෝලීය පරාසයේ `$x` සහ ස්ථානීය පරාසයේ `$x` යනු වෙනස් විචල්‍ය දෙකකි.

Ex.

```
<?php

$globalVar = "I'm a global variable";

function testScope() {

    $localVar = "I'm a local variable";
    echo $localVar; // Accessible
    echo "<br>";

    // echo $globalVar;
    // Error: Undefined variable

}

testScope();
echo $globalVar; // Accessible

// echo $localVar;
// Error: Undefined variable

?>
```

I'm a local variable
I'm a global variable

The global Keyword

To access global variables within a function, the `global` keyword can be used.

ශ්‍රිතයක් තුළදී ගෝලීය විචල්‍යයන් වෙත ප්‍රවේශ වීමට `global` මූල පදය භාවිතා කළ හැක.

```
<?php
$globalVar = "I'm a global variable";

function testScope() {

    global $globalVar;
    echo $globalVar; // Accessible
}

testScope();

?>
```

I'm a global variable

The \$GLOBALS Array

The `$GLOBALS` array is a superglobal associative array in PHP that contains references to all variables available in the global scope of the script.

`$GLOBALS` අරාච් යනු කේතයේ ගෝලීය පරාමිති අර්ථ දැක්වීම් සියලුම විචල්‍යයන් සඳහා යොමු අඩංගු PHP හි සුපිරි ගෝලීය associative array වකි.

It provides a way to access global variables from anywhere in the script, including within functions and methods where global variables are not normally accessible.

එය සාමාන්‍යයෙන් ගෝලීය විචල්‍යයන් වෙත ප්‍රවේශ විය නොහැකි ශ්‍රිත සහ methods ඇතුළුව, කේතයේ ඕනෑම තැනක සිට ගෝලීය විචල්‍ය වෙත ප්‍රවේශ වීමට මාර්ගයක් සපයයි.

NOTE: The keys in the `$GLOBALS` array are the names of the global variables, and the values are the contents of those variables.

`$GLOBALS` අරාවේ keys යනු ගෝලීය විචල්‍යවල නම් වන අතර එම keys වල අගයන් යනු විචල්‍යවල අන්තර්ගතය වේ.

It allows functions and methods to access and modify global variables without explicitly using the `global` keyword.

`global` මූල පදය භාවිතා නොකර ගෝලීය විචල්‍යයන් වෙත ප්‍රවේශ වීමට සහ වෙනස් කිරීමට එය ශ්‍රිත සහ methods වලට ඉඩ දෙයි.

```
<?php

$globalVar = "I'm a global variable";

function testGlobalAccess() {

    echo $GLOBALS['globalVar'];
}

testGlobalAccess();

?>
```

I'm a global variable

In this example, the function `testGlobalAccess` accesses the global variable `$globalVar` using the `$GLOBALS` array.

මෙම උදාහරණයේදී, `testGlobalAccess` ශ්‍රිතය `$GLOBALS` අරාච් භාවිතයෙන් `$globalVar` ගෝලීය විචල්‍යයට ප්‍රවේශ වේ.

Modifying Global Variables

```
<?php

$globalVar = "I'm a global variable";

function modifyGlobal() {

    echo $GLOBALS['globalVar'];
    echo "<br>";
    $GLOBALS['globalVar'] = "I've
    been modified!";
}

modifyGlobal();
echo $globalVar;

?>
```

I'm a global variable
I've been modified!

Using \$GLOBALS with Multiple Global Variables

ගෝලීය විචල්‍යයන් කිහිපයක් සමඟ \$GLOBALS භාවිතා කිරීම

```
<?php

$var1 = "First variable";
$var2 = "Second variable";

function showGlobals() {

    echo $GLOBALS['var1'];
    echo "<br>";
    echo $GLOBALS['var2'];
}

showGlobals();

?>
```

First variable
Second variable

In this example, the function showGlobals accesses multiple global variables using the \$GLOBALS array.

මෙම උදාහරණයේදී, showGlobals ශ්‍රිතය \$GLOBALS අරාච්ඡා භාවිතයෙන් බහු ගෝලීය විචල්‍යයන් වෙත ප්‍රවේශ වේ.

Avoiding Conflicts with Local Variables

ස්ථානීය විචල්‍යයන් සමඟ ඇති වන ගැටලු වළක්වා ගැනීම

The \$GLOBALS array helps prevent conflicts between local and global variables by providing a clear distinction and access mechanism.

\$GLOBALS අරාච්ඡා පැහැදිලි වෙනසක් සහ ප්‍රවේශ යාන්ත්‍රණයක් ලබා දීමෙන් ස්ථානීය සහ ගෝලීය විචල්‍යයන් අතර ගැටලු වළක්වා ගැනීමට උපකාරී වේ.

```
<?php

$var = "Global variable";

function testConflict() {
    $var = "Local variable";
    echo $var; echo "<br>";
    echo $GLOBALS['var'];
}

testConflict();

?>
```

Local variable
Global variable

In this example, the function testConflict demonstrates the distinction between a local variable and a global variable with the same name, using the \$GLOBALS array to access the global variable.

මෙම උදාහරණයේ දී, testConflict ශ්‍රිතය මඟින් ගෝලීය විචල්‍යයට ප්‍රවේශ වීමට \$GLOBALS අරාච්ඡා භාවිතා කරමින් ස්ථානීය විචල්‍යය නම සහිත විචල්‍යයක් සහ ගෝලීය විචල්‍යයක් අතර වෙනස පෙන්නුම් කරයි.

Default Parameters

You can specify default values for function parameters.

ඔබට ශ්‍රිත පරාමිති සඳහා පෙරනිමි අගයන් නියම කළ හැක.

If a new argument for a parameter with a default value is not provided during the function call, that default value will be used. ශ්‍රිත ඇමතුම අතරතුර පෙරනිමි අගයක් සහිත පරාමිතියකට එකක් සපයා නොමැති නම්, එම පෙරනිමි අගය භාවිතා වේ.

Ex.

```
<?php
function greet($name = "Guest") {
    echo "Hello, $name!";
}

greet();
greet("Alice");
?>
```

Hello, Guest!
Hello, Alice!

Recursive Functions

A recursive function is a function that calls itself.

පුනරාවර්තන ශ්‍රිතයක් යනු තමන් විසින්ම කැඳවා ගන්නා ශ්‍රිතයකි.

Recursion is useful for tasks that can be divided into similar subtasks.

සමාන උප කාර්යයන් වලට බෙදිය හැකි කාර්යයන් සඳහා පුනරාවර්තන ප්‍රයෝජනවත් වේ.

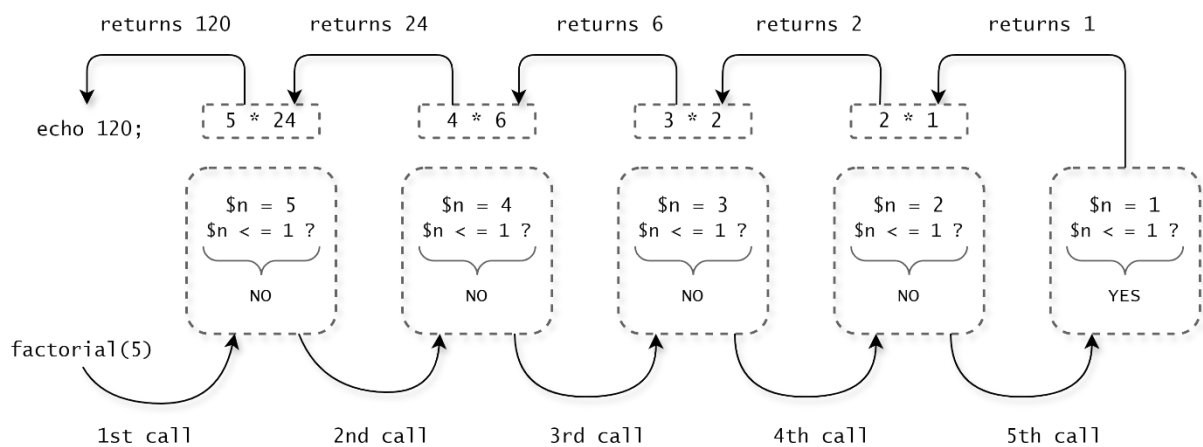
Ex. Factorial Calculation

```
<?php
function factorial($n) {
    if ($n <= 1) {
        return 1;
    } else {
        return $n * factorial($n - 1);
    }
}

echo factorial(5);
?>
```

120

How Recursive functions work



Consider each box as a function call

Commonly used built-in functions

1. String Functions

strlen()

Returns the length of a string.
තන්තුවක දිග ලබා දෙයි.

```
<?php
echo strlen("Hello");
?>
```

5

str_replace()

Replaces all occurrences of a search string with a replacement string.
සෙවුම් තන්තුවක සියලුම සිදුවීම් ප්‍රතිස්ථාපන තන්තුවකින් ප්‍රතිස්ථාපනය කරයි.

```
<?php
$txt = "Hello world!";
echo str_replace("Hello", "Hi",
$txt);
?>
```

Hi world!

strpos()

Finds the position of the first occurrence of a substring in a string.
තන්තුවක ඇති උප තන්තුවක පළමු සිදුවීමේ පිහිටීම සොයා ගනී.

```
<?php
echo strpos("Hello world!", "world");
?>
```

6

substr()

Returns a portion of a string.
තන්තුවක කොටසක් ලබා දෙයි.

```
<?php
echo substr("Hello world!", 6, 5);
?>
```

world

2. Array Functions

count()

Counts all elements in an array or something in an object.
අරාවක හෝ වස්තුවක ඇති සියලුම මූලාංගවල ගණන් කරයි.

```
<?php
$arr = array(1, 2, 3);
echo count($arr);
?>
```

3

array_merge()

Merges one or more arrays.
අරා එකක් හෝ කිහිපයක් එකතුවීම කරයි.

```
<?php

$array1 = array(1,2,3);
$array2 = array(4,5,6);
$result = array_merge($array1,
$array2);
var_dump($result);
?>
```

array (size=6)

```
0 => int 1
1 => int 2
2 => int 3
3 => int 4
4 => int 5
5 => int 6
```


array_push()

Pushes one or more elements onto the end of an array.

මූලාවයව එකක් හෝ කිහිපයක් අරාවක අවසානයට තල්ලු කරයි.

```
<?php
$arr = array("apple", "banana");
array_push($arr, "orange", "grape");
var_dump($arr);

?>
```

array (size=4)

```
0 => string 'apple' (length=5)
1 => string 'banana' (length=6)
2 => string 'orange' (length=6)
3 => string 'grape' (length=5)
```

array_keys()

Returns all the keys or a subset of the keys of an array.

අරාවක සියලුම යතුරු හෝ යතුරුවල උප කුලකයක් ආපසු ලබා දෙයි.

```
<?php
$arr = array("a" => "apple",
            "b" => "banana");
var_dump(array_keys($arr));

?>
```

array (size=2)

```
0 => string 'a' (length=1)
1 => string 'b' (length=1)
```

3. Mathematical Functions

abs()

Returns the absolute value of a number.

සංඛ්‍යාවක නිරපේක්ෂ අගය ලබා දෙයි.

```
<?php
echo abs(-4.2);

?>
```

4.2

round()

Rounds a floating point number.

ඉපිලෙන ලක්ෂ්‍ය සංඛ්‍යාවක් වටයයි.

```
<?php
echo round(3.6);

?>
```

4

rand()

Generates a random integer.

සසම්භාවී පූර්ණ සංඛ්‍යාවක් ජනනය කරයි.

```
<?php
echo rand(1, 10);
// Outputs: a random number
// between 1 and 10

?>
```

sqrt()

Returns the square root of a number.

සංඛ්‍යාවක වර්ගමූලය ලබා දෙයි.

```
<?php
echo sqrt(16);

?>
```

4

Other commonly used functions

අනෙකුත් බහුලව භාවිතා ශ්‍රිත

isset()

The `isset()` function in PHP is used to determine if a variable is set and is not NULL.

(ජ්‍යෙෂ්ඨ) හි ඇති `isset()` ශ්‍රිතය විචල්‍යයක් සකසා තිබේද සහ NULL නොවේද යන්න තීරණය කිරීමට භාවිතා කරයි.

Returns true if the variable exists and is not NULL.

විචල්‍යය පවතින්නේ නම් සහ NULL නොවේ නම් සත්‍ය වේ.

Returns false if the variable does not exist or is NULL.

විචල්‍යය නොමැති නම් හෝ NULL නම් අසත්‍ය වේ.

```
<?php
$variable = "Hello, World!";
if (isset($variable)) {
    echo "Variable is set";
} else {
    echo "Variable is not set";
}
?>
```

Variable is set

empty()

Determines whether a variable is empty.

විචල්‍යයක් හිස් ද යන්න තීරණය කරයි.

```
<?php
$var = "";
echo empty($var);
?>
```

1

print_r()

Prints human-readable information about a variable.

විචල්‍යයක් ගැන මිනිසුන්ට කියවිය හැකි තොරතුරු මුද්‍රණය කරයි.

```
<?php
$arr = array("apple", "banana");
print_r($arr);
?>
```

Array ([0] => apple [1] => banana)

var_dump()

Dumps information about a variable.

විචල්‍යයක් පිළිබඳ තොරතුරු ඩම්ප් කරයි.

```
<?php
$var = "Hello";
var_dump($var);
?>
```

string 'Hello' (length=5)

die() or exit()

Outputs a message and terminates the current script.

පණිවිඩයක් ප්‍රතිදානය කර වත්මන් කේතය එක අවසන් කරයි.

```
<?php
die("Script terminated");
?>
```

Script terminated

implode()

The implode() function (also known as join()) is used to join elements of an array into a single string.

අරාමක මූලාවයව තනි තන්තුවකට සම්බන්ධ කිරීමට implode() ශ්‍රිතය (join()) ලෙසද හැඳින්වේ) භාවිතා වේ.

Syntax

```
string implode(string $separator,  
array $array)
```

Example

```
<?php  
$array = ['apple', 'banana'];  
$string = implode(", ", $array);  
echo $string;  
?>
```

explode()

The explode() function is used to split a string by a specified delimiter into an array.

හිඟ්විත පරිසීමකයක් මගින් තන්තුවක් අරාමකට බෙදීමට explode() ශ්‍රිතය භාවිතා කරයි.

Syntax

```
array explode(string $separator,  
string $string)
```

Example

```
<?php  
$string = "apple, banana, cherry";  
$array = explode(", ", $string);  
print_r($array);  
?>
```

```
Array ([0] => apple [1] => banana  
[2] => cherry)
```

static keyword

The static keyword in PHP allows a local variable to retain its value across multiple calls to that function.

PHP හි මූල පදය ස්ථානීය විචල්‍යයකට එම ශ්‍රිතයට ඇමතුම් කිහිපයක් හරහා එහි අගය රඳවා ගැනීමට ඉඩ සලසයි.

This means that the value of the variable is memorized between function calls.

මෙයින් අදහස් කරන්නේ විචල්‍යයේ අගය ශ්‍රිත ඇමතුම් අතර මතක තබා ගෙන ඇති බවයි.

```
<?php  
function counter() {  
    static $count = 0;  
    $count++;  
    echo $count;  
}
```

```
counter();  
counter();  
counter();  
?>
```

```
1  
2  
3
```

Handling HTML forms with PHP

PHP සමගින් HTML පෝරම හැසිරවීම

HTML forms are used to collect user input and submit it to a server for processing.

HTML පෝරම භාවිතා කරනුයේ පරිශීලක ආදාන චක්‍ර කුර විය සැකසීම සඳහා සේවාදායකයක පරිගණකයකට යැවීමටයි.

HTML Form

First Name :

Last Name :

Date of Birth :

Email id :

Mobile Number :

They are essential for creating interactive web applications, such as login forms, search boxes, and data entry forms.

පිවිසුම් පෝරම, සෙවුම් පෙට්ටි සහ දත්ත ඇතුළත් කිරීමේ පෝරම වැනි අන්තර්ක්‍රියාකාරී වෙබ් යෙදුම් නිර්මාණය කිරීම සඳහා ඒවා අත්‍යවශ්‍ය වේ.

Benefits of using HTML forms

HTML පෝරම භාවිතා කිරීමේ වාසි

- **Access Control**
ප්‍රවේශ පාලනය

Forms act as an intermediary, preventing users from directly interacting with the database.

පෝරම අතරමැදියෙකු ලෙස ක්‍රියා කරයි, පරිශීලකයින් දත්ත සම්ප්‍රදාය සමඟ සෘජුව අන්තර් ක්‍රියා කිරීම වළක්වයි.

This reduces the risk of unauthorized access and malicious activities such as SQL injection.

මෙමගින් SQL injection වැනි අනවසර ප්‍රවේශයන් සහ අනිෂ්ට ක්‍රියාකාරකම් වල අවදානම අඩු කරයි.

- **Validation**
වලංගු කිරීම

HTML forms and server-side scripts can validate and sanitize user input to ensure that it meets specific criteria before it is entered into the database, preventing harmful data from being stored.

HTML පෝරම සහ සේවාදායක පාර්ශ්ව කේත වලට පරිශීලක ආදානය වලංගු කිරීමට සහ විය දත්ත සම්ප්‍රදායට ඇතුළු කිරීමට පෙර විය නිශ්චිත නිර්ණායක සපුරාලන බවට සහතික කිරීමට සහ හානිකර දත්ත ගබඩා කිරීම වළක්වයි.

- **Consistent Data Entry**
ස්ථාවර දත්ත ඇතුළත් කිරීම

Forms can enforce specific formats and data types, ensuring that the data entered by users is consistent and structured correctly.

පරිශීලකයන් විසින් ඇතුළත් කරන ලද දත්ත ස්ථාවර සහ නිවැරදිව ව්‍යුහගත බව සහතික කරමින්, පෝරමවලට නිශ්චිත ආකෘති සහ දත්ත වර්ග බලාත්මක කළ හැක.

- **Error Checking**
දෝෂ පරීක්ෂා කිරීම

Forms can provide immediate feedback to users about incorrect or incomplete data, reducing the likelihood of errors in the database.

දත්ත සම්පූර්ණයේ දෝෂ ඇතිවීමේ සම්භාවිතාව අඩු කරමින්, වැරදි හෝ අසම්පූර්ණ දත්ත පිළිබඳව පරිශීලකයින්ට ක්ෂණික ප්‍රතිචාර පෝරම මගින් ලබා දිය හැක.

- **User Experience**
පරිශීලක අත්දැකීම්

Forms can be designed to be user-friendly, making it easier for users to enter data correctly.

පෝරම පරිශීලක-හිතකාමී ලෙස නිර්මාණය කළ හැකි අතර, පරිශීලකයින්ට දත්ත නිවැරදිව ඇතුළත් කිරීම පහසු කරයි.

They can include dropdowns, checkboxes, and other input types to simplify data entry. දත්ත ඇතුළත් කිරීම සරල කිරීම සඳහා ඒවාට dropdowns, checkboxes සහ වෙනත් ආදාන වර්ග ඇතුළත් කළ හැක.

Forms can include instructions, tooltips, and validation messages to guide users through the data entry process.

දත්ත ඇතුළත් කිරීමේ ක්‍රියාවලිය හරහා පරිශීලකයින්ට මඟ පෙන්වීම සඳහා උපදෙස්, මෙවලම් ඉඟි සහ වලංගු කිරීමේ පණිවිඩ පෝරමවලට ඇතුළත් කළ හැක.

- **Controlled Data Flow**
පාලිත දත්ත ප්‍රවාහය

Forms allow developers to control the flow of data, ensuring that it is processed and stored correctly.

පෝරම මගින් සංවර්ධකයින්ට දත්ත ගලායාම පාලනය කිරීමට ඉඩ සලසයි, එය නිවැරදිව සකසන ලද සහ ගබඩා කර ඇති බව සහතික කරයි.

This includes logging submissions, sending confirmation emails, or triggering other processes.

Submit කිරීම් log කිරීම, තහවුරු කිරීමේ ඊමේල් යැවීම හෝ වෙනත් ක්‍රියාවලි සිදුකිරීම මෙයට ඇතුළත් වේ.

- **Data Preprocessing**
දත්ත පෙර සැකසුම්

Before data is entered into the database, it can be preprocessed in various ways, such as formatting dates, normalizing text, or aggregating data.

දත්ත සම්පූර්ණයට දත්ත ඇතුළත් කිරීමට පෙර, එය දින ආකෘතිකරණය කිරීම, පෙළ සාමාන්‍යකරණය කිරීම හෝ දත්ත එකතු කිරීම වැනි විවිධ ආකාරවලින් පෙර සැකසිය හැක.

Basic Structure of an HTML Form

HTML පෝරමයක මූලික ව්‍යුහය

<form> tag

An HTML form is defined using the <form> element, which can contain various types of input elements.

විවිධ ආකාරයේ ආදාන මූලද්‍රව්‍ය අඩංගු විය හැකි <form> මූලාවයවය භාවිතයෙන් HTML පෝරමයක් අර්ථ දැක්වනු ලබයි.

A form can contain various types of input elements, buttons, and other elements that enable users to submit data to a web server.

පෝරමයක විවිධ ආකාරයේ ආදාන මූලාවයව, බොත්තම් සහ වෙබ් සේවාදායකයකට දත්ත ඉදිරිපත් කිරීමට පරිශීලකයින්ට හැකි වෙනත් මූලාවයව අඩංගු විය හැක.

☐ Label ☒ Option A

☒ Label ☐ Option B

☐

Write an excerpt (optional)

Example:

```
<body>
<form action="submit.php" method="post" autocomplete="on">
  <!-- Form elements go here -->
</form>
</body>
```

<form> Attributes

action

Specifies the URL to which the form data will be submitted.

පෝරම දත්ත ඉදිරිපත් කරන URL වික සඳහන් කරයි.

method

Specifies the HTTP method to use when submitting the form (get or post).

පෝරමය ඉදිරිපත් කිරීමේදී භාවිතා කළ යුතු HTTP ක්‍රමය සඳහන් කරයි (get හෝ post).

(Action & method attributes will be further discussed later.)

(Action සහ method යන උපලක්ෂණ පසුව තවදුරටත් සාකච්ඡා කෙරේ.)

autocomplete

Specifies whether the form should have autocomplete on or off.

පෝරමය ස්වයං සම්පූර්ණ කිරීම සක්‍රිය හෝ අක්‍රිය කළ යුතුද යන්න සඳහන් කරයි.

<input> tag

The <input> element is the most widely used form element, supporting various input types such as text, password, email, and more.

<input> මූලාශ්‍රය යනු පෙළ, මුරපද, විද්‍යුත් තැපෑල සහ තවත් බොහෝ ආදාන වර්ග සඳහා සහය දක්වන වඩාත් බහුලව භාවිතා වන පෝරම මූලාශ්‍රය වේ.

<input> Attributes

type

Specifies the type of input (Ex. text, password, email, radio, checkbox, submit, etc.). (will be discussed later)

ආදාන වර්ගය සඳහන් කරයි (උදා. පෙළ, මුරපද, විද්‍යුත් තැපෑල, රේඩියෝ බොත්තම්, පිරික්සුම් කොටු, submit කිරීම ආදිය).

name

Specifies the name of the input, which is used to identify the form data after submission.

Submit කිරීමෙන් පසු පෝරම දත්ත හඳුනා ගැනීමට භාවිතා කරන ආදානයේ නම සඳහන් කරයි.

value

Specifies the initial value of the input. ආදානයේ ආරම්භක අගය නියම කරයි.

placeholder

Provides a hint to the user about what to enter in the input.

ආදානය තුළ ඇතුළත් කළ යුතු දේ ගැන පරිශීලකයාට ඉඟියක් සපයයි.

Email

name@example.com 

Placeholder

required

Specifies that the input must be filled out before submitting the form.

පෝරමය කිරීමට පෙර ආදානය පිරවීම අනිවාර්ය බව සඳහන් කරයි.

Name:*

! Please fill out this field

Please enter the following information to create your account.

First name *

Last name *

Email address *

Password *

Confirm password *

Required

readonly

Specifies that the input field is read-only.

ආදාන ක්ෂේත්‍රය කියවීමට පමණක් බව සඳහන් කරයි.

First Name*

Last Name*

Email*

Phone

disabled

Disables the input field.

ආදාන ක්ෂේත්‍රය අක්‍රිය කරයි.

Value (disabled)

id

Specifies a unique identifier for the input. ආදානය සඳහා අනන්‍ය හඳුනාගැනීමක් නියම කරයි.

The name attribute in detail

උපලක්ෂණය විස්තරාත්මකව

The name attribute in HTML form elements is a fundamental component for web forms, playing a crucial role in the form submission process, server-side data handling, and client-side scripting.

HTML පෝරම මූලාශ්‍රවල name උපලක්ෂණය වෙබ් පෝරම සඳහා මූලික අංගයක් වන අතර, පෝරමය submit කිරීමේ ක්‍රියාවලියේදී, සේවාදායක අන්තර්ගතයේ දත්ත හැසිරවීමේදී, සහ සේවාලාභී පාර්ශ්වයේ ස්ක්‍රිප්ට් කිරීමේදී තීරණාත්මක කාර්යභාරයක් ඉටු කරයි.

What is the name attribute?

name උපලක්ෂණය යනු කුමක්ද?

The name attribute is used to specify a unique identifier for form elements.

පෝරම මූලාශ්‍රවල සඳහා අනන්‍ය හඳුනාගැනීමක් නියම කිරීමට name උපලක්ෂණය භාවිතා කරයි.

This identifier is used when data is submitted to the server, allowing the server-side script to retrieve and process the form data.

මෙම හඳුනාගැනීමකරකය පෝරම දත්ත ලබා ගැනීමට සහ සැකසීමට සේවාදායක අන්තර්ගතයේ කේතයට ඉඩ සලසමින් සේවාදායකය වෙත දත්ත submit කරන විට භාවිතා කරනු ලැබේ.

Syntax

```
<input type="text" name="username">
```

Importance and Functions of the name Attribute

When a form is submitted, the browser sends the form data as **key-value** pairs to the server.

පෝරමයක් submit කළ විට, අතරික්ෂුව මගින් පෝරම දත්ත යතුරු-අගය යුගල ලෙස සේවාදායකයට යවයි.

The name attribute of each form element acts as the key in these pairs. The value entered by the user is the value in the key-value pair.

එක් එක් පෝරම මූලාශ්‍රවල name උපලක්ෂණය මෙම යුගලවල යතුර ලෙස ක්‍රියා කරයි. පරිශීලකයා විසින් ඇතුළත් කරන ලද අගය යතුරු-අගය යුගලයේ අගය වේ.

Example:

```
<body>
<form action="submit.php" method="post">
  <input type="text" name="username" placeholder="Username">
  <input type="password" name="password" placeholder="Password">
  <button type="submit">Login</button>
</form>
</body>
```

When this form is submitted, the data sent to the server might look like this:

මෙම පෝරමය submit කළ විට, සේවාදායකය වෙත යවන ලද දත්ත මෙලෙස පෙනිය හැකිය:

username=johndoe&password=securepassword

<label> tag

The <label> element defines a label for an <input> element.

<label> මූලාවයවය <input> මූලාවයවයක් සඳහා ලේඛනයක් අර්ථ දක්වයි.

Clicking on the label will focus/select the corresponding input.

ලේඛනය මත ක්ලික් කිරීමෙන් අදාළ ආදානය අවධානය යොමු කරයි/තෝරනු ඇත.

for Attribute

Specifies which form element a label is bound to by referring to the id of that particular element.

ලේඛනයක් බැඳී ඇත්තේ කුමන පෝරම මූලාවයවයටද යන්න එම මූලාවයවයේ අනන්‍ය හඳුනාගැනීම වෙත යොමු කරමින් සඳහන් කරයි.

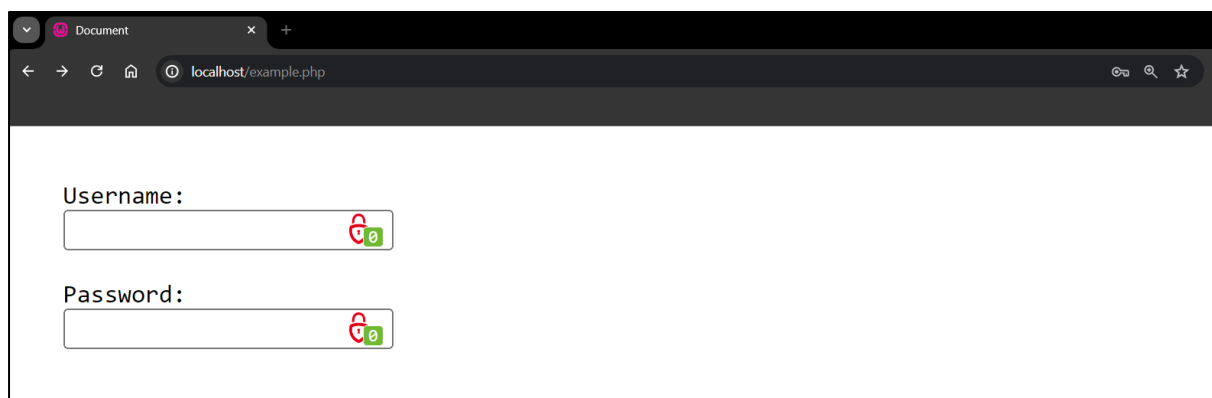
Example:

```
<body>
<form> <pre>

  <label for="username">Username:</label>
  <input type="text" id="username" name="u_name">

  <label for="password">Password:</label>
  <input type="password" id="password" name="pwd">

</pre> </form>
</body>
```



<fieldset> and <legend> tags

<fieldset>

The <fieldset> tag is used to create a grouping of form elements.

පෝරම මූලාවයව සමූහයක් සෑදීමට <fieldset> උසුලනය භාවිතා කරයි.

It helps in organizing the form elements into logical sections, making the form easier to understand and use.

විය පෝරමය මූලාවයවය තාර්කික කොටස් වලට සංවිධානය කිරීමට උපකාරී වන අතර, පෝරමය තේරුම් ගැනීමට සහ භාවිතා කිරීමට පහසු කරයි.

<legend>

The <legend> tag is used to provide a caption or title for the <fieldset>.

<fieldset> සඳහා විකක් හෝ මාතෘකාවක් සැපයීමට <legend> උසුලනය භාවිතා වේ.

It should be the first element inside the <fieldset> and gives a brief description of the group of form elements.

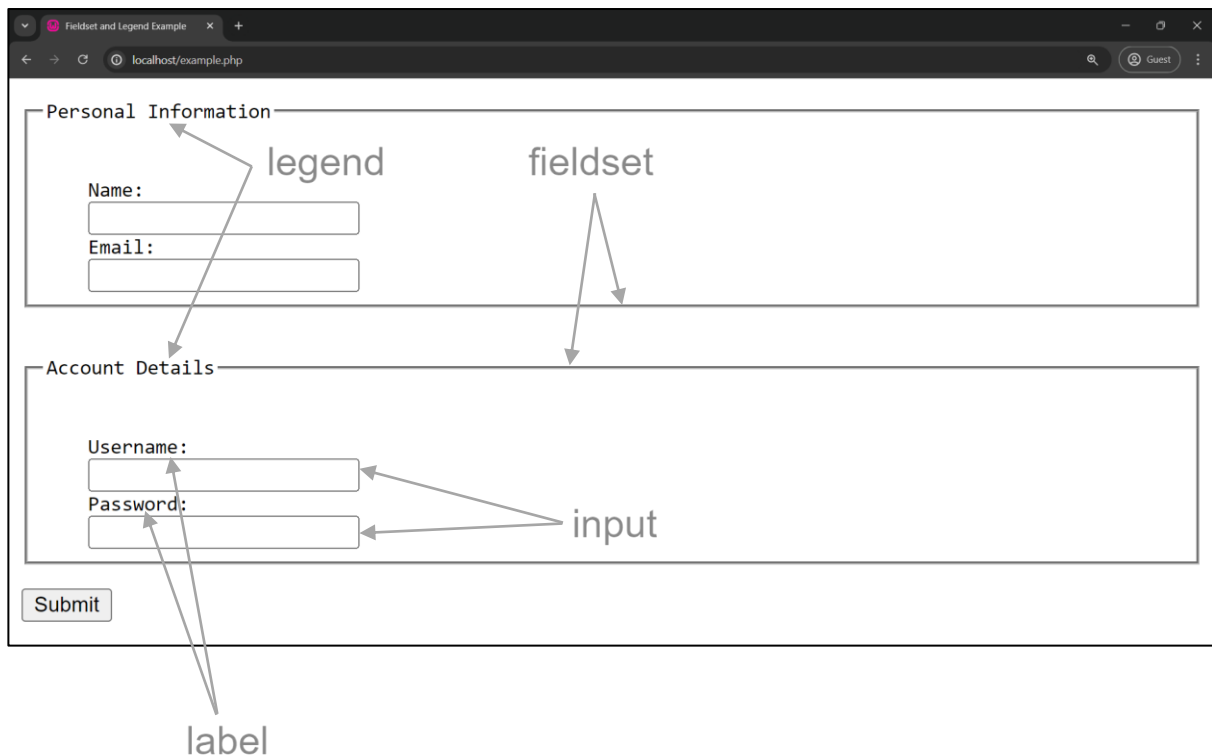
විය <fieldset> තුළ ඇති පළමු මූලාවයවය විය යුතු අතර පෝරම මූලාවයව පිළිබඳ කෙටි විස්තරයක් ලබා දෙයි

Example: (<fieldset> & <legend>)

```
<!DOCTYPE html>
<html>
<head>
  <title>Fieldset and Legend Example</title>
</head>
<body>
<form> <pre>
<fieldset>
  <legend>Personal Information</legend>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name">
  <label for="email">Email:</label>
  <input type="email" id="email" name="email">
</fieldset>

<fieldset>
  <legend>Account Details</legend>
  <label for="username">Username:</label>
  <input type="text" id="username" name="username">
  <label for="password">Password:</label>
  <input type="password" id="password" name="password">
</fieldset>
<input type="submit" value="Submit">

</pre> </form>
</body>
</html>
```



Input Types

ආදාන වර්ග

HTML forms use various input types to collect different kinds of user data.

HTML පෝරම විවිධ ආකාරයේ පරිශීලක දත්ත රැස් කිරීමට විවිධ ආදාන වර්ග භාවිතා කරයි.

1. Text (<input type="text">)

Used for single-line text input.

තනි පේළියේ පෙළ ආදානය සඳහා භාවිතා වේ.

Username:

2. Password (<input type="password">)

Used for password input, hides the characters.

මුරපද ආදානය සඳහා භාවිතා කරයි, අනුලක්ෂණ සහවයි.

Password:

3. Email (<input type="email">)

Used for email addresses, includes validation.

විද්‍යුත් තැපෑල ලිපින සඳහා භාවිත කෙරේ, වලංගුකරණය ඇතුළත් වේ.

Email:

Email:

❗ Please include an '@' in the email address. 'abcm@gmail.com' is missing an '@'.

4. Number (<input type="number">)

Used for numeric input, includes validation for numbers


සංඛ්‍යාත්මක ආදානය සඳහා භාවිත කෙරේ, සංඛ්‍යා සඳහා වලංගු කිරීම ඇතුළත් වේ


Number:



5. Date (<input type="date">)

Used for date input, provides a date picker.


දින ආදානය සඳහා භාවිතා කරයි, දින තෝරාගැනීම සපයයි.

Date: 

Date: 

July 2024 ▾  


Su	Mo	Tu	We	Th	Fr	Sa
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10
Clear		Today				


Date: 

6. Time (<input type="time">)


Used for time input, provides a time picker.

කාල ආදානය සඳහා භාවිතා කරයි, කාල තෝරාගැනීම සපයයි.

Time: 

Time: 

06	49	PM
07	50	AM
08	51	
09	52	
10	53	
11	54	
12	55	

Time: 

7. Checkbox (`<input type="checkbox">`)

Checkboxes allow users to select multiple options from a set.

Checkboxes මගින් පරිශීලකයින්ට කට්ටලයකින් විකල්ප කිහිපයක් තෝරා ගැනීමට ඉඩ සලසයි.

Each checkbox operates independently, even if they share the same name attribute සෑම checkbox එකක්ම එකම name උපලක්ෂණය බෙදා ගත්තද එවා ස්වාධීනව ක්‍රියා කරයි

Subscribe to newsletter: ☐

Receive updates: ☒

Get special offers: ☒

Submit

HTML code for the above page (`<html>`, `<body>` are not mentioned here)

ඉහත පිටුව සඳහා HTML කේතය

```
<form action="submit.php" method="post">
  <label for="newsletter">Subscribe to newsletter:</label>
  <input type="checkbox" id="newsletter" name="preferences[]"
  value="newsletter">

  <label for="updates">Receive updates:</label>
  <input type="checkbox" id="updates" name="preferences[]"
  value="updates">

  <label for="offers">Get special offers:</label>
  <input type="checkbox" id="offers" name="preferences[]"
  value="offers">

  <button type="submit">Submit</button>
</form>
```

The name attribute is set to preferences[], creating an array of selected values when the form is submitted.

name උපලක්ෂණය preferences[] වෙත සකසා ඇත, පෝරමය submit කරන විට තෝරාගත් අගයන්ගෙන් සමන්විත අරායක් නිර්මාණය කරයි.

Server-side processing (PHP)

```
<?php
$preferences = $_POST['preferences']; // $preferences is an array
foreach ($preferences as $preference) {
    echo htmlspecialchars($preference) . "<br>";
}
?>
```

8. Radio Buttons

(`<input type="radio">`)

Radio buttons allow users to select only one option from a group.

Radio buttons මගින් පරිශීලකයින්ට කණ්ඩායමකින් එක් විකල්පයක් පමණක් තෝරා ගැනීමට ඉඩ සලසයි.

Radio buttons with the same name attribute are grouped together, meaning only one button in the group can be selected at a time.

එකම name උපලක්ෂණය සහිත Radio buttons එකට කාණ්ඩගත කර ඇත, එනම් එම කාණ්ඩයෙන් එක් බොත්තමක් පමණක් වරකට තෝරා ගත හැකිය.

Male: ☒

Female: ☐

Other: ☐

Submit

HTML code for the above page

ඉහත පිටුව සඳහා HTML කේතය

```
<form action="submit.php" method="post">
  <label for="male">Male:</label>
  <input type="radio" id="male" name="gender" value="male">

  <label for="female">Female:</label>
  <input type="radio" id="female" name="gender" value="female">

  <label for="other">Other:</label>
  <input type="radio" id="other" name="gender" value="other">

  <button type="submit">Submit</button>
</form>
```

The name attribute is set to gender, grouping the radio buttons together.

Radio buttons එකට සමූහගත කරමින් name උපලක්ෂණය gender ලෙසට සකසා ඇත.

Server-Side processing (PHP)

```
<?php
$gender = $_POST['gender']; // $gender contains the selected value
echo "Selected gender: " . htmlspecialchars($gender);
?>
```

Only the selected value will be submitted and accessible.

තෝරාගත් අගය පමණක් submit වන අතර එයට පමණක් ප්‍රවේශ විය හැකිය.

9. File (<input type="file">)

Used for file uploads.

ගොනු උඩුගත කිරීම් සඳහා භාවිතා වේ.

File: Choose File No file chosen

10. Hidden (<input type="hidden">)

Used for hidden data, not visible to the user but sent with the form.

සැඟවුණු දත්ත සඳහා භාවිතා වේ, පරිශීලකයාට නොපෙනෙන නමුත් පෝරමය සමඟ යවනු ලැබේ.

11. Submit (<input type="submit">)

Used for submitting the form data.

පෝරම දත්ත submit කිරීම සඳහා භාවිතා වේ.

First Name: John

Last Name: William

submit

12. Reset (<input type="reset">)

Used for resetting the form to its initial values.

පෝරමය එහි ආරම්භක අගයන් වෙත නැවත සැකසීමට භාවිතා කරයි.

First Name:

Last Name:

submit

Reset

13. Color (<input type="color">)

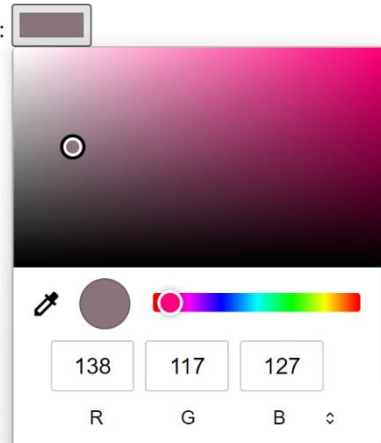
Used for color input, provides a color picker.

වර්ණ ආදානය සඳහා භාවිතා කරයි, වර්ණ තෝරා ගැනීමක් සපයයි.

Pick a color:



Pick a color:



14. Range (<input type="range">)

Used for numeric input within a range, provides a slider.

පරාසයක් තුළ සංඛ්‍යාත්මක ආදානය සඳහා භාවිතා වේ, ස්ලයිඩරයක් සපයයි.

Range:



15. URL (<input type="url">)

Used for URL input, includes validation.

URL ආදානය සඳහා භාවිතා කරයි, වලංගු කිරීම ඇතුළත් වේ.

<select> and <option> tags

The <select> and <option> HTML tags are used to create drop-down lists in forms. පෝරමවල පහළට වැටෙන ලැයිස්තු සෑදීමට <select> සහ <option> යන උසුලන භාවිතා කරයි.

These elements allow users to select one or more options from a predefined list. මෙම මූලාශ්‍රය පරිශීලකයින්ට පූර්ව නිශ්චිත ලැයිස්තුවකින් විකල්ප එකක් හෝ කිහිපයක් තෝරා ගැනීමට ඉඩ සලසයි.

<select> Tag

The <select> tag creates a drop-down list. It is used to define the list and can contain multiple <option> tags inside it. <select> උසුලනය පහළට වැටෙන ලැයිස්තුවක් සාදයි. එය ලැයිස්තුව නිර්වචනය කිරීමට භාවිතා කරන අතර එය තුළ <option> උසුලන කිහිපයක් අඩංගු විය හැක.

Attributes of <select> <select> හි උපලක්ෂණ

name

Specifies the name of the drop-down list, which is sent to the server when the form is submitted. පෝරමය submit කරන විට සේවාදායකය වෙත යවනු ලබන පහළට වැටෙන ලැයිස්තුවේ නම සඳහන් කරයි.

id

Unique identifier for the <select> element, useful for linking labels. <select> මූලාශ්‍රය සඳහා අනන්‍ය හඳුනාගැනීම, ලේබල සම්බන්ධ කිරීම සඳහා ප්‍රයෝජනවත් වේ.

size

Specifies the number of visible options in the drop-down list. පහළට වැටෙන ලැයිස්තුවේ පෙනෙන විකල්ප ගණන නියම කරයි.

multiple

Allows multiple selections if set. By default, only one option can be selected. සකසා ඇත්නම් බහු තේරීම් වලට ඉඩ දෙයි. පෙරනිමියෙන්, තෝරා ගත හැක්කේ එක් විකල්පයක් පමණි.

disabled

Disables the drop-down list, preventing user interaction. පරිශීලක අන්තර්ක්‍රියා වළක්වමින් පතන ලැයිස්තුව අක්‍රිය කරයි.

<option> Tag

The <option> tag defines an option in the drop-down list. <option> උසුලනය පහළට වැටෙන ලැයිස්තුවේ විකල්පයක් නිර්වචනය කරයි.

It is placed inside the <select> tag. එය <select> උසුලනය තුළ තබා ඇත.

Attributes of <option>

value

Specifies the value to be sent to the server when the option is selected. If omitted, the text content of the <option> is sent. විකල්පය තෝරාගත් විට සේවාදායකය වෙත යැවිය යුතු අගය සඳහන් කරයි. මෙය ඉවත් කර ඇත්නම්, <option> හි පෙළ අන්තර්ගතය යවනු ලැබේ.

selected

Pre-selects the option when the page loads. පිටුව පූරණය වන විට විකල්පය පෙර-තෝරා ගනී.

disabled

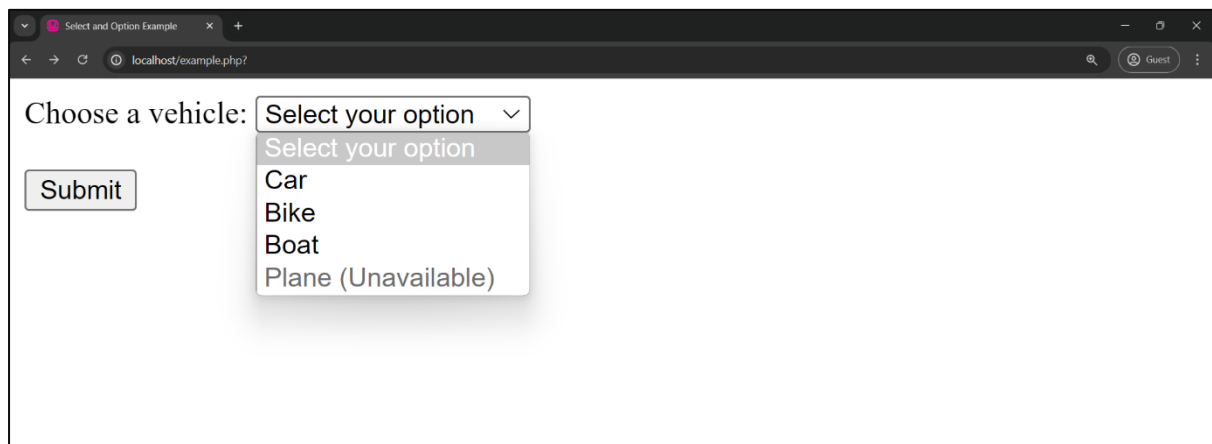
Disables the option, preventing user selection. විකල්පය අක්‍රිය කරයි, පරිශීලක තේරීම වළක්වයි.

Example.

```
<html>
<head>
  <title> Select and Option example </title>
</head>
<body>
<form action="/submit_form" method="post">

  <label for="vehicle-select">Choose a vehicle:</label>
  <select name="vehicle" id="vehicle-select" required>
    <option value="" disabled selected>Select your option</option>
    <option value="car">Car</option>
    <option value="bike">Bike</option>
    <option value="boat">Boat</option>
    <option value="plane" disabled>Plane (Unavailable)</option>
  </select>

</form>
</body>
</html>
```



Special Note on the name Attribute

The name attribute in the <select> tag specifies the key used to reference the selected value when the form data is submitted.

<select> උසුලනය තුළ ඇති උපලක්ෂණය පෝරම දත්ත submit කරන විට තෝරාගත් අගය යොමු කිරීමට භාවිතා කරන යතුර සඳහන් කරයි.

This attribute is crucial for server-side processing.

සේවාදායක අන්තර්ගත සැකසුම් සඳහා මෙම උපලක්ෂණය ඉතා වැදගත් වේ.

Action attribute

Action උපලක්ෂණය

The action attribute of the <form> tag specifies the URL where the form data should be sent when the form is submitted.

<form> උපලක්ෂණයේ action උපලක්ෂණය පෝරමය submit කරන විට පෝරම දත්ත යැවිය යුතු URL එක සඳහන් කරයි.

It defines the location of the server-side script or application that will process the form data.

එය පෝරම දත්ත සකසන, සේවාදායක අන්තර්ගතය හෝ යෙදුමේ පිහිටීම නිර්වචනය කරයි.

This attribute is essential for determining how the form data is handled after submission.

Submit කිරීමෙන් පසු පෝරම දත්ත හසුරුවන්නේ කෙසේද යන්න තීරණය කිරීම සඳහා මෙම උපලක්ෂණය අත්‍යවශ්‍ය වේ.

Method attribute

Method උපලක්ෂණය

When an HTML form is submitted, the data can be sent to the server using either the GET or POST method.

HTML පෝරමයක් submit කළ විට, දත්ත GET හෝ POST ක්‍රමය භාවිතයෙන් සේවාදායකයට යැවිය හැක.

Each method has its own characteristics and use cases.

මෙම ක්‍රම දෙකටම වෙන් වූ ලක්ෂණ සහ භාවිත අවස්ථා ඇත.

GET Method

The GET method appends form data to the URL specified in the action attribute of the form.

GET ක්‍රමය පෝරමයේ action ගුණාංගයේ දක්වා ඇති URL හි පෝරමයට ආදානය කළ දත්ත එකතු කරයි.

This means the data is visible in the browser's address bar.

මෙයින් අදහස් කරන්නේ එම දත්ත අතරක්සුවේ ලිපින තීරුවේ දිස්වන බවයි

Characteristics of the GET Method

GET ක්‍රමයේ ලක්ෂණ

Data is appended to the URL as query parameters, making it visible and shareable.

දෘශ්‍යමාන සහ බෙදාගත හැකි ලෙස දත්ත විමසුම් පරාමිති ලෙස URL වෙත එකතු කරයි.

The length of the URL is limited, so GET requests are suitable for sending small amounts of data.

URL හි දිග සීමිත වේ, එබැවින් GET ඉල්ලීම් කුඩා දත්ත ප්‍රමාණයක් යැවීම සඳහා සුදුසු වේ.

GET requests can be cached by browsers and intermediary devices.

GET ඉල්ලීම් අතරක්සු සහ අතරමැදි උපාංග මගින් නිභිත කළ හැක.

Repeated requests (Ex. refreshing the page) with the same URL will have the same effect.

එකම URL සමඟ නැවත නැවත ඉල්ලීම් (උදා. පිටුව refresh කිරීම) මගින් එකම බලපෑමක් ඇති කරයි.

Example:

```
<!DOCTYPE html>
<html>
<head></head>
<body>

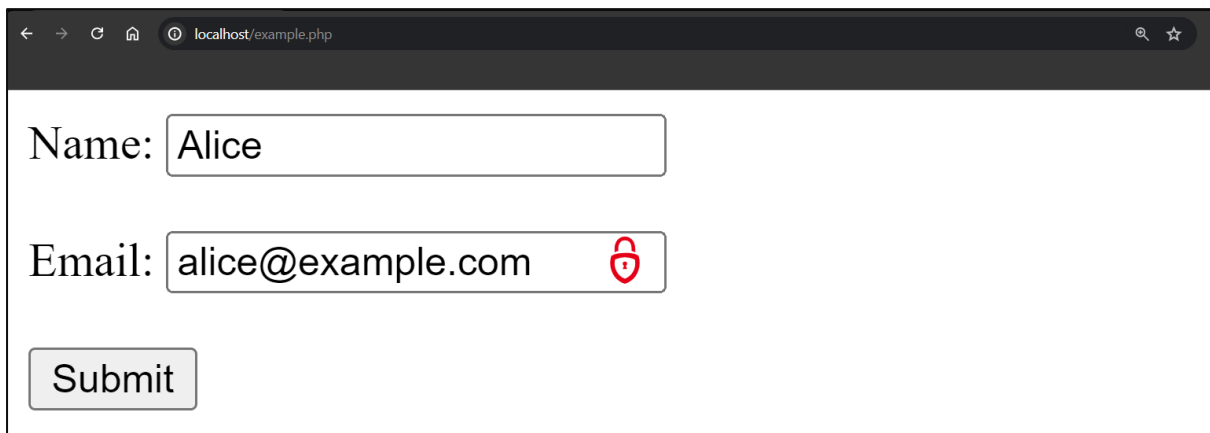
  <form action="process.php" method="get">

    <label for="name">Name:</label>
    <input type="text" id="name" name="name"><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email"><br><br>

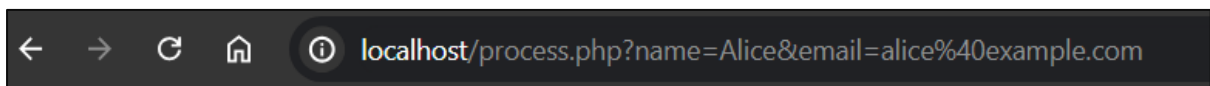
    <input type="submit" value="Submit">

  </form>
</body>
</html>
```

A screenshot of a web browser window. The address bar shows 'localhost/example.php'. The page content includes a form with two input fields: 'Name:' followed by a text input containing 'Alice', and 'Email:' followed by an email input containing 'alice@example.com'. A red padlock icon is visible next to the email input. Below the inputs is a 'Submit' button.

If the user enters "Alice" for the name and "alice@example.com" for the email, the URL will look like this upon submission:

පරිශීලකයා නම සඳහා "Alice" සහ විද්‍යුත් තැපෑල සඳහා "alice@example.com" ඇතුළු කරන්නේ නම්, Submit කිරීමේදී URL එක මෙලෙස දිස් වේ:

A screenshot of a web browser window showing the URL in the address bar after the form submission. The URL is 'localhost/process.php?name=Alice&email=alice%40example.com'.

Use Cases for GET

GET වල භාවිත

Searching for information (Ex. search queries).

තොරතුරු සෙවීම (උදා. සෙවුම් විමසුම්).

Retrieving data that does not change the state of the server.

සේවාදායකයේ තත්ත්වය වෙනස් නොවන දත්ත ලබා ගැනීම.

Bookmarking and sharing URLs with specific parameters.

නිශ්චිත පරාමිතින් සහිත URL පිටු සලකුණු කිරීම සහ බෙදාගැනීම සඳහා.

Used to retrieve non-sensitive data

(not suitable for things like passwords)

සංවේදී නොවන දත්ත ලබා ගැනීමට භාවිතා කරයි (මුරපද වැනි දේ සඳහා සුදුසු නොවේ)

POST Method

The POST method sends form data in the body of the HTTP request, not in the URL.

POST ක්‍රමය මගින් පෝරම දත්ත යවනුයේ HTTP ඉල්ලීමේ අන්තර්ගතය තුළය. URL තුළ නොවේ.

This makes the data invisible in the browser's address bar.

මෙමගින් බ්‍රවුසරයේ ලිපිත තීරුවේ දත්ත නොපෙනී යයි.

Characteristics of the POST Method

POST ක්‍රමයේ ලක්ෂණ

Data is sent in the body of the HTTP request, not visible in the URL.

දත්ත HTTP ඉල්ලීමේ අන්තර්ගතය තුළ යවනු ලැබේ, URL හි නොපෙනේ.

POST requests can handle larger amounts of data compared to GET requests.

GET ඉල්ලීම්වලට සාපේක්ෂව POST ඉල්ලීම්වලට විශාල දත්ත ප්‍රමාණයක් හැසිරවිය හැක.

POST requests are not cached by default.

POST ඉල්ලීම් පෙරහිමියෙන් නිතින නොවේ.

POST requests cannot be bookmarked because the data is not in the URL.

දත්ත URL හි නොමැති නිසා POST ඉල්ලීම් පිටු සලකුණු කළ නොහැක.

Repeated requests can have different effects (e.g., submitting a form multiple times).

නැවත නැවත කරන ලද ඉල්ලීම්වලට විවිධ බලපෑම් ඇති විය හැක (උදා: පෝරමයක් කිහිප වතාවක් කිරීම).

Example:

```
<!DOCTYPE html>
<html>
<head></head>
<body>

  <form action="process.php" method="post">

    <label for="name">Name:</label>
    <input type="text" id="name" name="name"><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email"><br><br>

    <input type="submit" value="Submit">

  </form>
</body>
</html>
```

Use Cases for POST

POST වල භාවිත

Submitting form data that changes the state of the server (Ex. creating a new user, processing payments).

සේවාදායකයේ තත්ත්වය වෙනස් කරන පෝරම දත්ත ඉදිරිපත් කිරීම (උදා: නව පරිශීලකයෙකු නිර්මාණය කිරීම, ගෙවීම් සකසීම).

Sending large amounts of data.

විශාල දත්ත ප්‍රමාණයක් යැවීම.

Uploading files.

ගොනු උඩුගත කිරීම.

Sending sensitive information (though HTTPS is recommended for security)

සංවේදී තොරතුරු යැවීම (ආරක්ෂාව සඳහා HTTPS නිර්දේශ කර තිබුණද)

Security Considerations

Validation

Always validate and sanitize form data to prevent security issues like SQL injection and XSS attacks.

SQL injection සහ XSS ප්‍රහාර වැනි ආරක්ෂක ගැටලු වැළැක්වීමට පෝරම දත්ත සැම විටම වලංගු කරන්න.

HTTPS

Use HTTPS to encrypt data in transit, especially when using GET for sensitive data.

නුවමාරුවේදී දත්ත සංකේතනය කිරීමට HTTPS භාවිතා කරන්න, විශේෂයෙන් සංවේදී දත්ත සඳහා GET භාවිතා කරන විට.

Data Exposure

Avoid using GET for sensitive data, as it can be exposed in browser history, server logs, and referral headers.

සංවේදී දත්ත සඳහා GET භාවිතා කිරීමෙන් වළකින්න, එය බ්‍රවුසර ඉතිහාසය, සේවාදායක ලොග සහ යොමු ශීර්ෂයන් තුළ නිරාවරණය විය හැක.

Superglobal variables

සුපිරි ගෝලීය විචල්‍යය

Superglobal variables in PHP are built-in global arrays that are always accessible, regardless of scope, meaning you can access them from any function, class, or file without needing to do anything special.

PHP හි ඇති සුපිරි ගෝලීය විචල්‍යයන් යනු පථය නොසලකා සෑම විටම ප්‍රවේශ විය හැකි ගෝලීය අරාමන් වේ, එනම් ඔබට විශේෂ කිසිවක් සිදු කිරීමට අවශ්‍ය නොවී ඕනෑම ශ්‍රිතයකින්, පන්තියකින් හෝ ගොනුවකින් ඒවාට ප්‍රවේශ විය හැක.

These superglobal variables are predefined by PHP and provide access to various kinds of data.

මෙම සුපිරි ගෝලීය විචල්‍යයන් PHP මගින් පූර්ව නිර්වචනය කර ඇති අතර විවිධ ආකාරයේ දත්ත වෙත ප්‍රවේශය සපයයි.

List of Superglobal Variables

සුපිරි ගෝලීය විචල්‍යය ලැයිස්තුව

- `$_GLOBALS`
- `$_SERVER`
- `$_GET`
- `$_POST`
- `$_FILES`
- `$_COOKIE`
- `$_SESSION`
- `$_REQUEST`
- `$_ENV`

We will only study about those are in the syllabus.

අප විසින් විෂය නිර්දේශයේ අඩංගු ඒවා පමණක් ඉගෙන ගනිමු.

`$_GLOBALS` (discussed earlier)

Contains a reference to every variable that is currently available in the global scope of the script.

කේතයේ ගෝලීය පථය තුළ දැනට පවතින සෑම විචල්‍යයක් සඳහාම යොමුවක් අඩංගු වේ.

Useful for accessing global variables from any point in the script.

කේතයේ ඕනෑම ස්ථානයක සිට ගෝලීය විචල්‍යයන් වෙත ප්‍රවේශ වීම සඳහා ප්‍රයෝජනවත් වේ.

```
<?php
$x = 10;
function test() {
    $GLOBALS['x']++;
}
test();
echo $x;
?>
```

11

`$_SERVER`

Contains information about headers, paths, and script locations. It is filled by the web server.

ශීර්ෂක, මාර්ග සහ කේත ස්ථාන පිළිබඳ තොරතුරු අඩංගු වේ. විය වෙබ් සේවාදායකය විසින් පුරවනු ඇත.

Common Elements

`$_SERVER['PHP_SELF']`

The filename of the currently executing script.

දැනට ක්‍රියාත්මක වන කේතයේ ගොනු නාමය

`$_SERVER['REQUEST_METHOD']`

The request method used to access the page (Ex. GET, POST).

පිටුවට ප්‍රවේශ වීමට භාවිතා කරන ඉල්ලුම් ක්‍රමය (උදා., GET, POST)

\$_SERVER['SERVER_NAME']

The name of the server host.
සේවාදායක සත්කාරකයේ නම.

\$_SERVER['HTTP_HOST']

The HTTP Host header.
HTTP සත්කාරක ශීර්ෂය.

\$_SERVER['HTTP_USER_AGENT']

The user agent string of the browser.
බ්‍රව්සරයේ පරිශීලක නියෝජිත පෙළ.

We can use the `var_dump()` to look
what's inside `$_SERVER`

`$_SERVER` තුළ ඇති දේ බැලීමට අපට
`var_dump()` භාවිතා කළ හැක

```
<?php
```

```
var_dump($_SERVER);
```

```
?>
```

```
D:\wamp64\www\example.php:3:
array (size=43)
  'HTTP_HOST' => string 'localhost' (length=9)
  'HTTP_CONNECTION' => string 'keep-alive' (length=10)
  'HTTP_CACHE_CONTROL' => string 'max-age=0' (length=9)
  'HTTP_SEC_CH-UA' => string '"Not/A)Brand";v="8", "Chromium";v="126", "Google Chrome";v="126"' (length=64)
  'HTTP_SEC_CH-UA-MOBILE' => string '?0' (length=2)
  'HTTP_SEC_CH-UA-PLATFORM' => string '"Windows"' (length=9)
  'HTTP_UPGRADE_INSECURE_REQUESTS' => string '1' (length=1)
  'HTTP_USER_AGENT' => string 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gec
  'HTTP_ACCEPT' => string 'text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image
  'HTTP_SEC_FETCH_SITE' => string 'same-origin' (length=11)
  'HTTP_SEC_FETCH_MODE' => string 'navigate' (length=8)
  'HTTP_SEC_FETCH_USER' => string '?1' (length=2)
  'HTTP_SEC_FETCH_DEST' => string 'document' (length=8)
  'HTTP_REFERER' => string 'http://localhost/example.php' (length=28)
  'HTTP_ACCEPT_ENCODING' => string 'gzip, deflate, br, zstd' (length=23)
  'HTTP_ACCEPT_LANGUAGE' => string 'en-US,en;q=0.9' (length=14)
  'PATH' => string 'C:\ProgramData\Oracle\Java\javapath;C:\Program Files\Common Files\Oracle\Java\javapath;C
  'SystemRoot' => string 'C:\WINDOWS' (length=10)
  'COMSPEC' => string 'C:\WINDOWS\system32\cmd.exe' (length=27)
  'PATHEXT' => string '.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC' (length=53)
  'WINDIR' => string 'C:\WINDOWS' (length=10)
  'SERVER_SIGNATURE' => string '<address>Apache/2.4.54 (Win64) PHP/8.0.26 mod_fcgid/2.3.10-dev Server at loc
  (length=101)
  'SERVER_SOFTWARE' => string 'Apache/2.4.54 (Win64) PHP/8.0.26 mod_fcgid/2.3.10-dev' (length=53)
  'SERVER_NAME' => string 'localhost' (length=9)
  'SERVER_ADDR' => string ':::1' (length=3)
  'SERVER_PORT' => string '80' (length=2)
  'REMOTE_ADDR' => string ':::1' (length=3)
  'DOCUMENT_ROOT' => string 'D:/wamp64/www' (length=13)
  'REQUEST_SCHEME' => string 'http' (length=4)
  'CONTEXT_PREFIX' => string '' (length=0)
  'CONTEXT_DOCUMENT_ROOT' => string 'D:/wamp64/www' (length=13)
  'SERVER_ADMIN' => string 'wampserver@wampserver.invalid' (length=29)
  'SCRIPT_FILENAME' => string 'D:/wamp64/www/example.php' (length=25)
  'REMOTE_PORT' => string '51139' (length=5)
  'GATEWAY_INTERFACE' => string 'CGI/1.1' (length=7)
  'SERVER_PROTOCOL' => string 'HTTP/1.1' (length=8)
  'REQUEST_METHOD' => string 'GET' (length=3)
  'QUERY_STRING' => string '' (length=0)
  'REQUEST_URI' => string '/example.php?' (length=13)
  'SCRIPT_NAME' => string '/example.php' (length=12)
  'PHP_SELF' => string '/example.php' (length=12)
  'REQUEST_TIME_FLOAT' => float 1721588066.2193
  'REQUEST_TIME' => int 1721588066
```

\$_GET

An associative array of variables passed to the current script via the URL parameters. URL පරාමිති හරහා වත්මන් කේතය වෙත යවන ලද විචල්‍යවලින් සමන්විත Associative array වකකි.

Used to collect form data after submitting an HTML form with the method="get".
method="get" පෝරමයක් submit කිරීමෙන් පසු පෝරම දත්ත රැස් කිරීමට භාවිතා කරයි.

A real-life example:

Consider we have an HTML form in a file called example.php.
example.php නම් ගොනුවක HTML පෝරමයක් ඇති බව සලකන්න.

Imagine we need to collect the data that user entered into the form and display that data back to the user.

පරිශීලකයා පෝරමයට ඇතුළු කළ දත්ත එකතු කර එම දත්ත නැවත පරිශීලකයාට පෙන්විය යුතු යැයි සිතන්න.

We can use another php file to collect this data. Let's call that file process.php.

මෙම දත්ත එකතු කිරීම සඳහා අපට වෙනත් php ගොනුවක් භාවිතා කළ හැකිය. අපි එම ගොනුව process.php ලෙස හඳුන්වමු.

example.php file

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Personal details</title>
</head>
<body>
<form method="get" action="process.php" style = "margin-left:100px;"> <pre>
<fieldset style = "width: 50%;">
  <legend>Data entry form</legend>
  <label for="userName">Username: </label>
  <input type="text" name="uname" id = "userName">

  <label for="email">Email: </label>
  <input type="email" name="email" id = "email">

  <label for="age">Age: </label>
  <input type="number" name="age" id = "age" min = "14" max = "65">

  <input type="radio" name="gender"> Male

  <input type="radio" name="gender"> Female

  <input type="checkbox" name="agree" value = "yes"> I agree!

  <input type="submit" value = "Show data">
</fieldset>
</pre></form> </body> </html>
```


The screenshot shows a web browser window with the address bar displaying 'localhost/example.php?'. The page content is a 'Data entry form' with the following elements:

- Username:** A text input field containing 'JohnWilliam100'.
- Email:** A text input field containing 'JohnWillion100@gmail.com'.
- Age:** A text input field containing '17'.
- Gender:** Two radio buttons, 'Male' (which is selected) and 'Female'.
- Agreement:** A checkbox labeled 'I agree!' which is currently unchecked.
- Action:** A button labeled 'Show data'.

When the user hits that show data button, we should be able to show the data that was entered by the user.

පරිශීලකයා show data බොත්තම එබූ විට, පරිශීලකයා විසින් ඇතුළත් කළ දත්ත පෙන්වීමට අපට හැකි විය යුතුය.

We have another file called process.php that is responsible for collecting and displaying this data back to the user.

අප සතුව process.php නමින් තවත් ගොනුවක් ඇත, එය මෙම දත්ත එකතු කිරීම සහ පරිශීලකයාට නැවත ප්‍රදර්ශනය කිරීම සඳහා වගකිව යුතුය.

Because we have used the GET method to send the data to the server, we can use the superglobal variable `$_GET` to collect that data.

සේවාදායකය වෙත දත්ත යැවීමට අප GET ක්‍රමය භාවිතා කර ඇති නිසා, එම දත්ත රැස් කිරීමට අපට `$_GET` යන සුපිරි ගෝලීය විචල්‍යය භාවිතා කළ හැකිය.

Since `$_GET` is an associative array, the keys of the key-value pairs are the values of the name attribute, while the values of the key-value pairs are the data entered by the user. `$_GET` යනු associative array එකක් බැවින්, යතුරු-අගය යුගලවල යතුරු name උපලක්ෂණය අගයන් වන අතර යතුරු-අගය යුගලවල අගයන් පරිශීලකයා විසින් ඇතුළත් කළ දත්ත වේ.

The PHP code to collect and display data back to the user will look like this.

දත්ත රැස් කිරීමට සහ ඒවා පරිශීලකයා වෙත නැවත ප්‍රදර්ශනය කිරීමට අදාළ PHP කේතය මේ ආකාරයෙන් පෙනෙනු ඇත.

process.php file

```
<?php

// Collecting the data using $_GET
$username = $_GET['uname'];
$email = $_GET['email'];
$age = $_GET['age'];
$gender = $_GET['gender'];

echo "<pre>"; // Using <pre> tags to make the web page look better:)

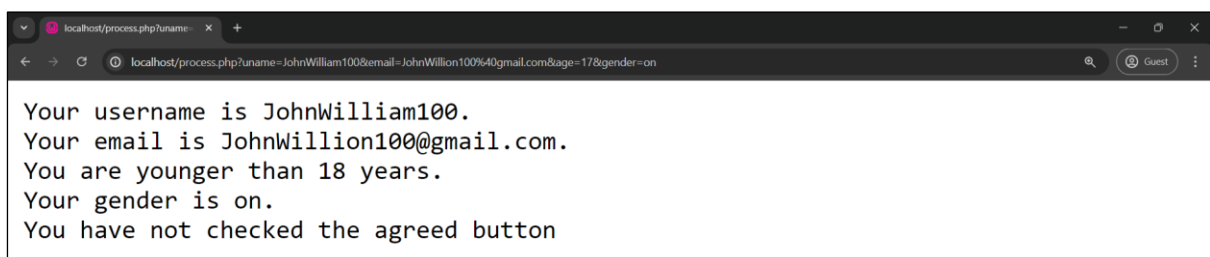
echo "Your username is $username. <br>";
echo "Your email is $email. <br>";

if($age > 18) {
    echo "You are older than 18 years.";
} elseif ($age == 18) {
    echo "You are exactly 18 years old.";
} else {
    echo "You are younger than 18 years.";
}
echo "<br>";
echo "Your gender is $gender. <br>";

// Checking if the user has checked the checkbox.
if (isset($_GET['agree'])) {
    echo "You have checked the agreed button";
} else {
    echo "You have not checked the agreed button";
}
echo "<pre>";
?>
```

When the show data button is pressed, the user will be redirected to the process.php file and displayed as follows:

දත්ත show data බොත්තම එබූ විට, පරිශීලකයා process.php ගොනුව වෙත හරවා යවා පහත පරිදි පෙන්වනු ඇත:



\$_POST

Used to collect form data after submitting an HTML form with the method="post".

method="post".සමඟ HTML පෝරමයක් කිරීමෙන් පසු පෝරම දත්ත රැස් කිරීමට භාවිතා කරයි.

In previous example, if we set the method attribute to post, we have to change the \$_GET to \$_POST (in process.php file) in order to collect the data entered by the user. There is no change in everything else. පෙර උදාහරණයේදී, අප method උපලක්ෂණය post ලෙස සකසන්නේ නම්, පරිශීලකයා විසින් ඇතුළත් කළ දත්ත විකත කිරීම සඳහා අපට \$_GET යන්න \$_POST (process.php ගොනුවේ) ලෙස වෙනස් කළ යුතුය. අනිකුත් සියල්ලේ කිසිදු වෙනසක් නොමැත.

Working with databases

Storing form data

පෝරම දත්ත ගබඩා කිරීම

After processing the form data, the next step is to store it in a database.

පෝරම දත්ත සැකසීමෙන් පසු, ඊළඟ පියවර වන්නේ විය දත්ත ගබඩාවක ගබඩා කිරීමයි.

PHP provides several extensions for database interactions, with PDO (PHP Data Objects) and MySQLi (MySQL Improved) being the most commonly used.

PDO (PHP Data Objects) සහ MySQLi (MySQL Improved) බහුලව භාවිතා වන දත්ත සමුදා අන්තර්ක්‍රියා සඳහා PHP දිගු කිහිපයක් සපයයි.

These extensions allow PHP to connect to various database management systems (DBMS), such as MySQL, PostgreSQL, and SQLite.

මෙම දිගුවන් MySQL, PostgreSQL, සහ SQLite වැනි විවිධ දත්ත සමුදා කළමනාකරණ පද්ධති (DBMS) වෙත සම්බන්ධ වීමට PHP හට ඉඩ සලසයි.

To connect to a database, the PHP script needs the database credentials, which typically include the hostname (or server address), database name, username, and password.

දත්ත සමුදායකට සම්බන්ධ වීමට, PHP කේතයට දත්ත සමුදායේ credentials අවශ්‍ය වේ, සාමාන්‍යයෙන් වියට සත්කාරක නාමය (හෝ සේවාදායක ලිපිනය), දත්ත සමුදා නාමය, පරිශීලක නාමය සහ මුරපදය ඇතුළත් වේ.

Using these credentials, PHP establishes a connection to the database server.

මෙම credentials භාවිතා කරමින්, PHP දත්ත සමුදා සේවාදායකයට සම්බන්ධතාවයක් ස්ථාපිත කරයි.

Once connected to the database, the PHP script prepares an SQL query to insert the processed form data into the appropriate table within the database.

දත්ත සම්ප්‍රදායට සම්බන්ධ වූ පසු, PHP කේතය විසින් සැකසූ පෝරම දත්ත දත්ත සම්ප්‍රදාය තුළ ඇති සුදුසු වගුවට ඇතුළත් කිරීමට SQL විමසුමක් සකස් කරයි.

This involves specifying the table name and the columns where the data should be stored.

මෙහි වගුවේ නම සහ දත්ත ගබඩා කළ යුතු තීරු නියම කිරීම ඇතුළත් වේ.

Prepared statements are often used in this step to enhance security and prevent SQL injection attacks.

මෙම පියවරේදී ආරක්ෂාව වැඩි දියුණු කිරීමට සහ SQL injection attacks වැළැක්වීමට පෙර සූදානම් කළ ප්‍රකාශ බොහෝ විට භාවිතා වේ.

The PHP script then executes the SQL query.

PHP කේතය පසුව SQL විමසුම ක්‍රියාත්මක කරයි.

If the execution is successful, the data is stored in the database, and the script can provide feedback to the user, such as a confirmation message or redirection to another page.

ක්‍රියාත්මක කිරීම සාර්ථක නම්, අදාළ දත්ත, දත්ත සම්ප්‍රදායේ ගබඩා කර ඇති අතර, තහවුරු කිරීමේ පණිවිඩයක් හෝ වෙනත් පිටුවකට හරවා යැවීමක් වැනි ප්‍රතිපෝෂණයක් කේතය මගින් පරිශීලකයාට ලබා දිය හැක.

If there is an error during the execution, the script should handle it gracefully by logging the error and informing the user without exposing sensitive information.

ක්‍රියාත්මක කිරීමේදී දෝෂයක් තිබේ නම්, කේතය විසින් දෝෂය ලොග් කිරීමෙන් සහ සංවේදී තොරතුරු හෙළිදරව් නොකර පරිශීලකයාට දන්වා විය අලංකාර ලෙස හැසිරවිය යුතුය.

Retrieving and Displaying Data

දත්ත ලබා ගැනීම සහ පිරිදර්ශනය කිරීම

In addition to storing data, PHP can also retrieve data from the database for various purposes.

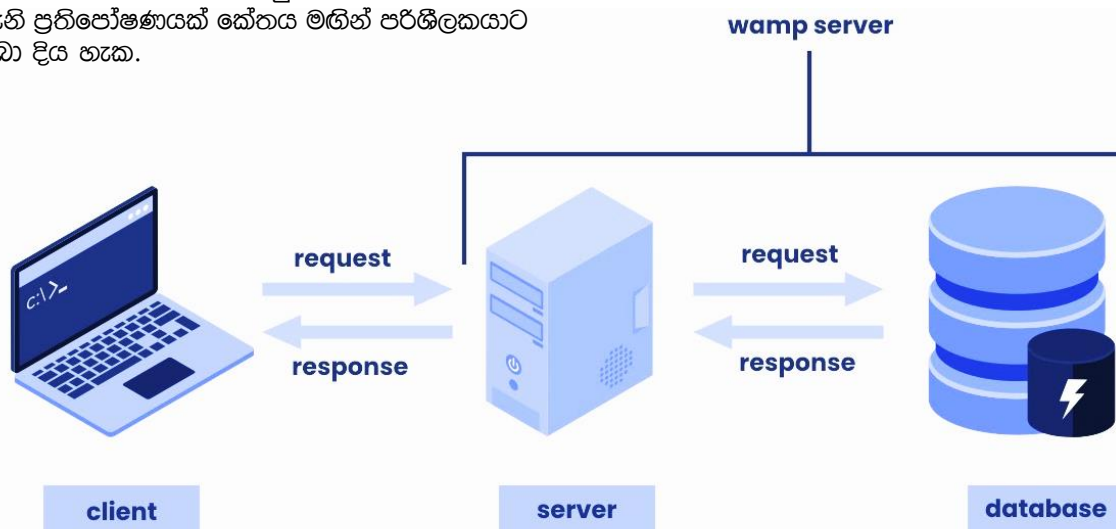
දත්ත ගබඩා කිරීමට අමතරව, PHP හට විවිධ අරමුණු සඳහා දත්ත ගබඩාවෙන් දත්ත ලබා ගැනීමටද හැක.

This involves executing SELECT queries and fetching the results.

SELECT විමසුම් ක්‍රියාත්මක කිරීම සහ ප්‍රතිඵල ලබා ගැනීම මෙයට ඇතුළත් වේ.

The retrieved data can then be displayed on web pages, providing dynamic content based on the information stored in the database.

දත්ත සම්ප්‍රදායේ ගබඩා කර ඇති තොරතුරු මත පදනම්ව ගතික අන්තර්ගතයන් ලබා දෙමින් ලබාගත් දත්ත පසුව වෙබ් පිටු මත පෙන්විය හැක.



Establishing database connection

දත්ත සමුදා සම්බන්ධතාවය ස්ථාපිත කිරීම

There are two primary approaches to establish a database connection in PHP.

PHP හි දත්ත සමුදා සම්බන්ධතාවයක් ස්ථාපිත කිරීම සඳහා මූලික ප්‍රවේශයන් දෙකක් තිබේ.

1. Structured approach (procedural)
2. Object-Oriented Programming (OOP) approach.

We will use the MySQLi extension for these examples.

මෙම උදාහරණ සඳහා අප MySQLi දිගුව භාවිතා කරමු

1. Structured approach (procedural)

```
<?php

// Database credentials
$servername = "localhost"; // 127.0.0.0 can also be used
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";

//Close the connection
mysqli_close($conn);
?>
```

Explanation

1. Database Credentials

\$servername

This variable stores the hostname or IP address of the database server. In most local development environments, this is localhost.

මෙම විචල්‍යය දත්ත සමුදා සේවාදායකයේ ධාරක නාමය හෝ IP ලිපිනය ගබඩා කරයි. බොහෝ ස්ථානීය සංවර්ධන පරිසරයන් තුළ, මෙය localhost වේ.

\$username

This is the username used to authenticate with the database. For WAMP servers, the default is usually root.

දත්ත සමුදාය සමඟ සත්‍යාපනය කිරීමට භාවිතා කරන පරිශීලක නාමය මෙයයි. WAMP සේවාදායකයන් සඳහා, පෙරනිමිය සාමාන්‍යයෙන් root වේ.

\$password

This is the password for the specified user. By default, for WAMP, it is usually an empty string ("").

මෙය යම් නිශ්චිත පරිශීලකයෙකුගේ මුරපදයයි. පෙරනිමියෙන්, WAMP සඳහා, එය සාමාන්‍යයෙන් හිස් තන්තුවකි ("").

\$dbname

This variable stores the name of the database you want to connect to. It must be created in MySQL beforehand.

මෙම විචල්‍යය ඔබට සම්බන්ධ වීමට අවශ්‍ය දත්ත සමුදායේ නම ගබඩා කරයි. එය කලින් MySQL හිදී නිර්මාණය කළ යුතුය.

2. Create Connection

mysqli_connect()

This function attempts to open a new connection to the MySQL server. It takes four parameters.

මෙම ශ්‍රිතය MySQL සේවාදායකයට නව සම්බන්ධතාවයක් විවෘත කිරීමට උත්සාහ කරයි. එය පරාමිති හතරක් ලබා ගනී.

The function returns a MySQLi link identifier on success (connection object), or false on failure.

ශ්‍රිතය මගින් සම්බන්ධතාව සාර්ථක වූ විට MySQLi සබැඳි හඳුනාගැනීමක් හෝ අසාර්ථක වූ විට ලෙස false ලබා දෙයි.

\$conn

This variable is used to store the result of the `mysqli_connect()` function which returns a connection object (if the connection is successful) or false (if the connection is unsuccessful).

සම්බන්ධතා වස්තුවක් (සම්බන්ධතාවය සාර්ථක නම්) හෝ false (සම්බන්ධතාවය අසාර්ථක නම්) ලෙස අගයක් ලබාදෙන `mysqli_connect()` ශ්‍රිතයේ ප්‍රතිඵලය ගබඩා කිරීමට මෙම විචල්‍යය භාවිතා කරයි.

3. Check Connection

if (!\$conn)

This condition checks if the connection failed. If \$conn is false, it means the connection was not successful.

මෙම කොන්දේසිය මගින් සම්බන්ධතාවය අසාර්ථක වී ඇත්දැයි පරීක්ෂා කරයි. \$conn අසත්‍ය නම්, එයින් අදහස් වන්නේ සම්බන්ධතාවය සාර්ථක නොවූ බවයි.

die()

This function prints a message and terminates the current script. If the connection fails, it prints "Connection failed: " followed by the error message.

මෙම ශ්‍රිතය මගින් පණිවිඩයක් මුද්‍රණය කර වත්මන් කේතය අවසන් කරයි. සම්බන්ධතාවය අසාර්ථක වුවහොත්, එය ("Connection failed: ") යන්න දෝෂ පණිවිඩයක් දෝශයද සමග මුද්‍රණය කරයි.

mysqli_connect_error()

This function returns a string description of the last connection error. It provides information about why the connection failed.

මෙම ශ්‍රිතය අවසාන සම්බන්ධතා දෝෂයේ විස්තරයක් ලබා දෙයි. සම්බන්ධතාවය අසාර්ථක වූයේ මන්දැයි යන්න විමගින් තොරතුරු සපයයි.

4. Closing the Connection

After completing the database operations, it is a good practice to close the connection to free up resources.

දත්ත සමුදා මෙහෙයුම් සම්පූර්ණ කිරීමෙන් පසුව, සම්පත් නිදහස් කිරීම සඳහා සම්බන්ධතාවය වසා දැමීම හොඳ පුරුද්දකි.

2. Object-Oriented Programming (OOP) approach.

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
$conn->close();
?>
```

Explanation

1. Create Connection

new mysqli

The new `mysqli` constructor in PHP is used to create an object/instance of the `MySQLi` class, which establishes a connection to a MySQL database.

PHP හි `new mysqli` යන්න `MySQL` දත්ත ගබඩාවකට සම්බන්ධයක් ඇති `MySQLi` පන්තියේ වස්තුවක්/අවස්ථාවක් නිර්මාණය කිරීමට භාවිතා කරයි.

The parameters passed to the constructor are the server name, username, password, and database name.

Constructor එක වෙත ලබා දී ඇති පරාමිති වන්නේ සේවාදායකයේ නම, පරිශීලක නාමය, මුරපදය සහ දත්ත සමුදා නාමයයි.

How new mysqli works

ක්‍රියා කරන ආකාරය

When the new `mysqli` constructor is called with valid parameters, it attempts to establish a connection to the MySQL database.

`new mysqli` constructor එක වලංගු පරාමිති සමඟින් ඇමතු වීම, එය `MySQL` දත්ත සමුදායට සම්බන්ධයක් ඇති කර ගැනීමට උත්සාහ කරයි.

The `$conn` variable holds an object of the `mysqli` class.

`$conn` විචල්‍යය `mysqli` පන්තියට අයිති වස්තුවක් දරයි.

If the connection is successful, this object contains properties and methods that can be used to interact with the database.

සම්බන්ධතාවය සාර්ථක නම්, මෙම වස්තුව තුළ දත්ත සමුදාය සමඟ අන්තර්ක්‍රියා කිරීමට භාවිතා කළ හැකි උපලක්ෂණ සහ methods අඩංගු වේ.

The object provides methods such as `query()`, `prepare()`, `close()`, etc., which are used to execute queries and manage the connection.

මෙම වස්තුව මගින් විමසුම් ක්‍රියාත්මක කිරීමට සහ සම්බන්ධතාවය කළමනාකරණය කිරීමට භාවිතා කරන `query()`, `prepare()`, `close()` වැනි methods සපයයි.

if there is an issue connecting to the MySQL server, it still creates an instance of the `mysqli` class.

MySQL සේවාදායකයට සම්බන්ධ වීමේ ගැටලුවක් තිබේ නම්, එය විසඳුවද `mysqli` පන්තියේ වස්තුවක් නිර්මාණය කරයි.

The `$conn` variable still holds an object of the `mysqli` class, but this object represents a failed connection.

`$conn` විචල්‍යය තවමත් `mysqli` පන්තියේ වස්තුවක් දරයි, නමුත් දැන් මෙම වස්තුව අසාර්ථක සම්බන්ධතාවයක් නියෝජනය කරයි.

The `connect_error` property of this object contains the error message describing why the connection failed.

මෙම වස්තුවේ `connect_error` උපලක්ෂණය තුළ සම්බන්ධතාවය අසාර්ථක වීමට හේතුව විස්තර කරන දෝෂ පණිවිඩය අඩංගු වේ.

2. Check Connection

```
if ($conn->connect_error)
```

`$conn->connect_error` is a property of the `mysqli` object that contains the error message if the connection fails.

`$conn->connect_error` යනු `mysqli` වස්තුවේ උපලක්ෂණයක් වන අතර එහි, සම්බන්ධතාවය අසමත් වුවහොත් දෝෂ පණිවිඩය අඩංගු වේ.

The if statement checks if there is a connection error.

if ප්‍රකාශය මගින් සම්බන්ධතා දෝෂයක් තිබේදැයි පරීක්ෂා කරයි.

```
$conn->connect_error
```

This property provides the description of the last connection error.

මෙම උපලක්ෂණය අවසාන සම්බන්ධතා දෝෂය පිළිබඳ විස්තරය සපයයි.

```
$conn->close()
```

Closes the database connection.

දත්ත සමුදා සම්බන්ධතාවය වසා දමයි.

Collecting form data and inserting that data into the database.

පෝරම දත්ත එකතු කර එම දත්ත දත්ත සම්ප්‍රදායට ඇතුළත් කිරීම.

HTML form

```
<!DOCTYPE html>
<html>
<head>
  <title>User Registration</title>
</head>
<body>
<form action="process.php" method="post">
  <fieldset>
    <legend> Register Here </legend>

    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="phone">Phone Number:</label>
    <input type="tel" id="phone" name="phone" required><br><br>

    <label>Gender:</label>
    <input type="radio" id="male" name="gender" value="Male" required>
    <label for="male">Male</label>
    <input type="radio" id="female" name="gender" value="Female" required>
    <label for="female">Female</label><br><br>

    <label>Interests:</label>
    <input type="checkbox" id="sports" name="interests[]" value="Sports">
    <label for="sports">Sports</label>
    <input type="checkbox" id="music" name="interests[]" value="Music">
    <label for="music">Music</label>
    <input type="checkbox" id="reading" name="interests[]" value="Reading">
    <label for="reading">Reading</label><br><br>

    <input type="submit" value="Submit">

  </fieldset>
</form>

</body>
</html>
```

Let's imagine the user fills the form as follows:
පරිශීලකයා විසින් පෝරමය පහත පරිදි පුරවන්නේ යැයි සිතමු:

Register Here

Name: John William

Email: JohnWilliam@gmail.com

Phone Number: 8011456012

Gender: ☒ Male ☐ Female

Interests: ☒ Sports ☒ Music ☒ Reading

Submit

Suppose we have a database table as below and want to insert form data into this table. අපට පහත පරිදි දත්ත සමුදා වගුවක් ඇති අතර පෝරමය දත්ත මෙම වගුවට ඇතුළු කිරීමට අවශ්‍ය බව සිතමු.

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(100) NOT NULL,
  phone VARCHAR(20) NOT NULL,
  gender VARCHAR(10) NOT NULL,
  interests VARCHAR(255)
);
```

When we collect data from users, there could be special characters or malicious inputs that can compromise the integrity and security of our SQL queries. අපි පරිශීලකයින්ගෙන් දත්ත රැස් කරන විට, අපගේ SQL විමසුම්වල අඛණ්ඩතාව සහ ආරක්ෂාව සම්මුතියට පත් කළ හැකි විශේෂ අක්ෂර හෝ අනිෂ්ට යෙදවුම් තිබිය හැක.

To prevent these issues, we need to sanitize and escape these inputs, ensuring they are treated as literal data rather than executable code.

මෙම ගැටළු වලක්වා ගැනීම සඳහා, ක්‍රියාත්මක කළ හැකි කේත වෙනුවට ඒවා වචනානුසාරයෙන් දත්ත ලෙස සලකන බව සහතික කරමින්, මෙම යෙදවුම් නාෂණය කර ගලවා ගත යුතුය.

mysqli_real_escape_string()

This is a built-in PHP function that prepares a string for use in an SQL query by escaping special characters.

මෙය PHP built-in ශ්‍රිතයක් වන අතර එය විශේෂ අක්ෂර වලින් ගැලවී SQL විමසුමක භාවිතා කිරීම සඳහා තත්ත්වයක් සූදානම් කරයි.

This ensures that the input data doesn't interfere with the query execution or compromise the database.

ආදාන දත්ත විමසුම් ක්‍රියාත්මක කිරීමට බාධා නොකරන බව හෝ දත්ත සමුදාය සම්මුතියට පත් නොවන බව මෙමගින් සහතික කරයි.

PHP Script to Process Form Data and Insert into Database (process.php file)

පෙරම දත්ත සැකසීමට සහ දත්ත සමුදායට ඇතුළු කිරීමට අදාළ PHP කේතය (process.php ගොනුව)

Procedural approach

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Collect and sanitize form data
$name = mysqli_real_escape_string($conn, $_POST['name']);
$email = mysqli_real_escape_string($conn, $_POST['email']);
$phone = mysqli_real_escape_string($conn, $_POST['phone']);
$gender = mysqli_real_escape_string($conn, $_POST['gender']);
$interests = isset($_POST['interests']) ? implode(", ", $_POST['interests']) : '';

// Prepare the SQL query
$sql = "INSERT INTO users (name, email, phone, gender, interests)
VALUES ('$name', '$email', '$phone', '$gender', '$interests')";

// Execute the query
if (mysqli_query($conn, $sql)) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}

// Close the connection
mysqli_close($conn);

?>
```

This is how the database table will look like after inserting the data entered by the user
පරිශීලකයා ආදානය කළ දත්ත ඇතුළත් කිරීමෙන් පසු දත්ත සමුදා වගුව මෙලෙස පෙනෙනු ඇත.

	id	first_name	last_name	email	phone	hire_date
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	John	Doe	john.doe@example.com	555-1234	2021-05-15
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	Jane	Smith	jane.smith@example.com	555-5678	2022-03-10
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	Emily	Johnson	emily.johnson@example.com	555-8765	2023-01-22

Explanation

HTML Form

This form collects the user's name, email, phone number, gender, and interests. මෙම පෝරමය පරිශීලකයාගේ නම, ඊමේල්, දුරකථන අංකය, ස්ත්‍රී පුරුෂ භාවය සහ රුචිකත්වයන් එකතු කරයි.

The interests[] checkboxes use an array notation to collect multiple values.

interests[] checkboxes ඔහු අගයන් එකතු කිරීම සඳහා අරා අංකනයක් භාවිතා කරයි.

PHP Script (process.php)

User input is sanitized using `mysqli_real_escape_string()` to prevent SQL injection. SQL injection වැළැක්වීමට `mysqli_real_escape_string()` භාවිතයෙන් පරිශීලක ආදානය නාෂණය කෙරේ.

Name, Email, Phone, and Gender:

These fields are straightforward text and radio inputs.

මෙම ක්ෂේත්‍ර සෘජු පෙළ සහ රේඩියෝ ආදාන වේ.

Interests

If the user has selected any interests, the values are joined into a single string using `implode(", ", $_POST['interests'])`.

පරිශීලකයා කිසියම් රුචිකත්වයක් තෝරාගෙන තිබේ නම්, එම අගයන් `implode(", ", $_POST['interests'])` භාවිතයෙන් තනි තනිවකට එකාබද්ධ කර ඇත.

SQL Query

The script prepares an SQL INSERT statement to add the collected data into the users table.

එකතු කරන ලද දත්ත වගුවට එක් කිරීමට කේතය විසින් SQL INSERT ප්‍රකාශයක් සකසයි.

Executing the Query

The script executes the query using `mysqli_query()`. කේතය `mysqli_query()` භාවිතයෙන් විමසුම ක්‍රියාත්මක කරයි.

If the query is successful, it outputs a success message.

විමසුම සාර්ථක නම්, එය සාර්ථක පණිවිඩයක් නිකුත් කරයි.

If there is an error, it outputs the error message.

දෝෂයක් තිබේ නම්, එය දෝෂ පණිවිඩයක් ප්‍රතිදානය කරයි.

OOP approach

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection using OOP approach
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Collect and sanitize form data
$name = $conn->real_escape_string($_POST['name']);
$email = $conn->real_escape_string($_POST['email']);
$phone = $conn->real_escape_string($_POST['phone']);
$gender = $conn->real_escape_string($_POST['gender']);
$interests = isset($_POST['interests']) ? implode(", ", $_POST['interests']) : '';

// Prepare the SQL query
$sql = "INSERT INTO users (name, email, phone, gender, interests)
VALUES ('$name', '$email', '$phone', '$gender', '$interests')";

// Execute the query
if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

// Close the connection
$conn->close();
?>
```

OOP approach executes the SQL query using the query() method.

OOP ප්‍රවේශය query() යන method එක භාවිතයෙන් SQL විමසුම ක්‍රියාත්මක කරයි.

Retrieving Data from an Existing Table

පවතින වගුවකින් දත්ත ලබා ගැනීම

Let's assume we have a MySQL table named employees with the following data.

පහත දත්ත සහිත employees නමින් MySQL වගුවක් අප සතුව ඇතැයි උපකල්පනය කරමු

id	first_name	last_name	phone	hire_date	email
1	John	Doe	555-1234	2021-05-15	john.doe@example.com
2	Jane	Smith	555-5678	2022-03-10	jane.smith@example.com
3	Emily	Johnson	555-8765	2023-01-22	emily.johnson@example.com

Connect to the Database

Establish a connection to the MySQL database using PHP. This involves providing the necessary credentials like server name, username, password, and database name.

PHP භාවිතයෙන් MySQL දත්ත සමුදායට සම්බන්ධතාවයක් ස්ථාපිත කරන්න. මෙයට සේවාදායක නාමය, පරිශීලක නාමය, මුරපදය සහ දත්ත සමුදා නාමය වැනි අවශ්‍ය අක්ෂරපත්‍ර සැපයීම ඇතුළත් වේ.

Prepare an SQL Query

Write an SQL SELECT query to fetch the data you need. For instance, you might want to retrieve all columns from the employees table.

ඔබට අවශ්‍ය දත්ත ලබා ගැනීමට SQL SELECT විමසුමක් ලියන්න. උදාහරණයක් ලෙස, ඔබට සේවක වගුවෙන් සියලුම තීරු ලබා ගැනීමට අවශ්‍ය විය හැකිය.

Execute the Query

Run the SQL query using PHP. This will send the query to the MySQL database and retrieve the results.

(PHP) භාවිතයෙන් SQL විමසුම ක්‍රියාත්මක කරන්න. මෙය MySQL දත්ත සමුදාය වෙත විමසුම යවා ප්‍රතිඵලය ලබා ගනී.

Process the Results

Check if the query returned any results. If so, fetch and process the data row by row. This usually involves iterating through the result set and displaying the data.

විමසුමෙන් කිසියම් ප්‍රතිඵලයක් ලැබේ ඇත්දැයි පරීක්ෂා කරන්න. එසේ ලැබේ ඇත් නම්, දත්ත පේළියෙන් පේළිය ගෙන සකසන්න. මෙය සාමාන්‍යයෙන් ප්‍රතිඵල කට්ටලය හරහා පුනරාවර්තනය කිරීම සහ දත්ත ප්‍රදර්ශනය කිරීම ඇතුළත් වේ.

Close the Connection

After processing the data, close the database connection to free up resources. දත්ත සැකසීමෙන් පසු, සම්පත් නිදහස් කිරීමට දත්ත සමුදා සම්බන්ධතාවය වසා දමන්න.

Retrieving Data

දත්ත ලබා ගැනීම

Procedural Approach

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// SQL query to fetch data
$sql = "SELECT * FROM employees";
$result = mysqli_query($conn, $sql);

// Check if there are any results
if (mysqli_num_rows($result) > 0) {

    // Output data of each row
    while($row = mysqli_fetch_assoc($result)) {

        echo "ID: " . $row["id"]. " - Name: " . $row["first_name"]. " " .
            $row["last_name"]. " - Email: " . $row["email"]. " - Phone: " .
            $row["phone"]. " - Hire Date: " . $row["hire_date"]. "<br>";
    }
} else {
    echo "0 results";
}

// Close connection
mysqli_close($conn);

?>
```

OOP approach

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection using OOP approach
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to fetch data
$sql = "SELECT * FROM employees";
$result = $conn->query($sql);

// Check if there are any results
if ($result->num_rows > 0) {

    // Output data of each row
    while($row = $result->fetch_assoc()) {
        echo "ID: " . $row["id"]. " - Name: " . $row["first_name"]. " " .
            $row["last_name"]. " - Email: " . $row["email"]. " - Phone: " .
            $row["phone"]. " - Hire Date: " . $row["hire_date"]. "<br>";
    }
} else {
    echo "0 results";
}

// Close connection
$conn->close();

?>
```


Explanation

Procedural

`mysqli_query($conn, $sql)`

In the procedural style, the `mysqli_query()` function is used to execute a query against the database.

Procedural ක්‍රමයේදී, දත්ත සම්ප්‍රදායට වෙත විමසුමක් ක්‍රියාත්මක කිරීමට `mysqli_query()` ශ්‍රිතය භාවිතා කරයි.

For successful SELECT, SHOW, DESCRIBE, or EXPLAIN queries, it returns a `mysqli_result` object representing the **result set**.

සාර්ථක SELECT, SHOW, DESCRIBE හෝ EXPLAIN විමසුම් සඳහා, එය ප්‍රතිඵල කට්ටලය නියෝජනය කරන `mysqli_result` වස්තුවක් ලබා දෙයි.

For successful INSERT, UPDATE, or DELETE queries, it returns TRUE.

සාර්ථක INSERT, UPDATE හෝ DELETE විමසුම් සඳහා, එය TRUE ලබා දෙයි.

If the query fails, it returns FALSE.

විමසුම අසාර්ථක වුවහොත්, එය FALSE ලෙස ලබා දෙයි.

`mysqli_num_rows($result) > 0`

`mysqli_num_rows()` is used to determine the number of rows in a result set returned by a SELECT query.

`mysqli_num_rows()` යන ශ්‍රිතය SELECT විමසුමකින් ආපසු ලැබෙන ප්‍රතිඵල කට්ටලයක පේළි ගණන තීරණය කිරීමට භාවිතා කරයි.

It takes the result set as an argument and returns an integer representing the number of rows.

එය ප්‍රතිඵල කට්ටලය තර්කයක් ලෙස ගෙන පේළි ගණන නියෝජනය කරන පූර්ණ සංඛ්‍යාවක් ලබා දෙයි.

`mysqli_fetch_assoc()`

For fetching data, `mysqli_fetch_assoc()` is used to retrieve a row from the result set as an associative array.

දත්ත ලබා ගැනීම සඳහා, ප්‍රතිඵල කට්ටලයෙන් පේළියක් associative array එකක් ලෙස ලබා ගැනීමට `mysqli_fetch_assoc()` භාවිතා කරයි.

This function returns the next row as an associative array where the keys are the column names of the table, and it returns FALSE when there are no more rows.

මෙම ශ්‍රිතය මගින් ඊළඟ පේළිය associative array එකක් ලෙස ලබා දෙයි, එහිදී යතුරු යනු වගුවේ තීරු නම් වන අතර තවත් පේළි නොමැති විට එය FALSE ලෙස ලබා දෙයි.

OOP

`$conn->query($sql)`

In the object-oriented style, the `query()` method of the `mysqli` class is used to perform a query against the database.

Object-oriented ක්‍රමයේදී, `mysqli` පන්තියේ `query()` විධි දත්ත සම්ප්‍රදාය වෙත විමසුමක් සිදු කිරීමට භාවිතා කරයි.

For successful SELECT, SHOW, DESCRIBE, or EXPLAIN queries, it returns a `mysqli_result` object representing the result set.

සාර්ථක SELECT, SHOW, DESCRIBE හෝ EXPLAIN විමසුම් සඳහා, එය ප්‍රතිඵල කට්ටලය නියෝජනය කරන `mysqli_result` වස්තුවක් ලබා දෙයි.

For successful INSERT, UPDATE, or DELETE queries, it returns TRUE.

සාර්ථක INSERT, UPDATE හෝ DELETE විමසුම් සඳහා, එය TRUE ලබා දෙයි.

If the query fails, it returns FALSE.

විමසුම අසාර්ථක වුවහොත්, එය FALSE ලෙස ලබා දෙයි.

`$result->num_rows > 0`

The `num_rows` property of a `mysqli_result` object is used to determine the number of rows in a result set returned by a SELECT query.

`mysqli_result` වස්තුවක `num_rows` උපලක්ෂණය SELECT විමසුමකින් ආපසු ලැබෙන ප්‍රතිඵල කට්ටලයක පේළි ගණන තීරණය කිරීමට භාවිතා කරයි.

It provides an integer representing the number of rows.

එය පේළි ගණන නියෝජනය කරන පූර්ණ සංඛ්‍යාවක් සපයයි.

`$result->fetch_assoc()`

For fetching data, the `fetch_assoc()` method of the `mysqli_result` class is used to retrieve a row from the result set as an associative array.

දත්ත ලබා ගැනීම සඳහා, ප්‍රතිඵල කට්ටලයෙන් පේළියක් associative array වකක් ලෙස ලබා ගැනීමට `mysqli_result` පන්තියේ `fetch_assoc()` method විධි භාවිතා කරයි.

This method returns the next row as an associative array where the keys are the column names of the table, and it returns FALSE when there are no more rows.

මෙම method විධි මගින් ඊළඟ පේළිය associative array වකක් ලෙස ලබා දෙයි, එහිදී යතුරු යනු වගුවේ තීරු නාම වන අතර තවත් පේළි නොමැති විට එය FALSE ලෙස ලබා දෙ

Other SQL operations using PHP

PHP භාවිත කරන අනෙකුත් SQL මෙහෙයුම්

Creating a table

වගුවක් නිර්මාණය කිරීම

```
<?php
// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to create table
$sql = "CREATE TABLE employees (
    id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL,
    phone VARCHAR(20),
    hire_date DATE
)";

// Execute the query
if ($conn->query($sql) === TRUE) {
    echo "Table employees created successfully";
} else {
    echo "Error creating table: " . $conn->error;
}

// Close the connection
$conn->close();
?>
```

Updating table data

වගු දත්ත සාවත්කාලීන කිරීම

```
<?php

// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to update data
$sql = "UPDATE employees SET email='john.newemail@example.com' WHERE id=1";

if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

// Close connection
$conn->close();

?>
```

Deleting table data

වගු දත්ත මකා දැමීම

```
<?php
// Database credentials
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "test_database";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// SQL query to delete data
$sql = "DELETE FROM employees WHERE id=3";

if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

// Close connection
$conn->close();
?>
```