

Treinamento Fundamentos de PHP

Turma 2023/01 – Aula 4/8



1

Semana 1



Aula 4 – Tratamento de erros

- Pilha de execução
- Erros e exceções
- Tratamento de exceções
- Lançamento de exceções
- KAHOOT

2

Recapitulando...



Na última aula vimos sobre:

- Funções definidas pelo usuário
 - Argumentos de funções
- Escopo
 - Global, local e superglobais
- Funções anônimas (closure)
- Arrow function
- Inclusão de arquivos php

- PHP na web
 - Subindo o servidor de desenvolvimento PHP
 - PHP com HTML
- Atividade 2: Alterar jogo TicTacToe
- Vamos complementar hoje
 - Mais sobre HTML
 - Como pegar informações do navegador para o servidor

Copyright© 2023 Accenture All Rights Reserved

3

3

PHP na web



Formulários HTML

- Usado para pegar entrada de dados do usuário
- O submit envia os dados do formulário para a url informada no "action" pelo método especificado (GET ou POST)

```

10 <h2>Formulário</h2>
11 <form action="pagina.php" method="post">
12 <label for="nomefuncionario"><b> Nome do funcionário</b></label>
13 <input type="text" name="nomefuncionario" id="nomefuncionario"><br>
14
15 <span><b> Alocação</b></span>
16 <label for="alocadosim">Sim</label>
17 <input type="radio" name="alocado" id="alocadosim">
18 <label for="alocadoNao">Não</label>
19 <input type="radio" name="alocado" id="alocadoNao"><br>
20
21 <span><b> Habilidades</b></span>
22 <label for="habilidadePHP">PHP</label>
23 <input type="checkbox" name="habilidades" id="habilidadePHP">
24 <label for="habilidadeJava">Java</label>
25 <input type="checkbox" name="habilidades" id="habilidadeJava"><br>
26
27 <label for="local"><b> Local</b></label>
28 <select name="local" id="local">
29 <option value="0" selected>Selecione</option>
30 <option value="Recife">Recife</option>
31 <option value="São Paulo">São Paulo</option>
32 <option value="Belo Horizonte">Belo Horizonte</option>
33 <option value="Campinas">Campinas</option>
34 </select>
35
36 <br><br>
37 <input type="submit" value="Enviar">
38 <input type="reset" value="Limpar formulário">
39 </form>

```

Copyright© 2023 Accenture All Rights Reserved

4

4

PHP na web



Formulários HTML

- Método GET
 - Os dados são passados pela URL
 - O comprimento da URL é limitado (cerca de 3.000 caracteres)
 - NÃO pode ser usado para dados confidenciais
- Método POST
 - Os dados são passados no corpo da solicitação HTTP
 - É mais seguro
 - Tamanho ilimitado

```

10 <h2>Formulário</h2>
11 <form action="pagina.php" method="post">
12   <label for="nomefuncionario"><b> Nome do funcionário</b></label>
13   <input type="text" name="nomefuncionario" id="nomefuncionario"><br>
14
15   <span><b> Alocação</b></span>
16   <label for="alocadoSim">Sim</label>
17   <input type="radio" name="alocado" id="alocadoSim">
18   <label for="alocadoNao">Não</label>
19   <input type="radio" name="alocado" id="alocadoNao"><br>
20
21   <span><b> Habilidades</b></span>
22   <label for="habilidadePHP">PHP</label>
23   <input type="checkbox" name="habilidades" id="habilidadePHP">
24   <label for="habilidadeJava">Java</label>
25   <input type="checkbox" name="habilidades" id="habilidadeJava"><br>
26
27   <label for="local"><b> Local</b></label>
28   <select name="local" id="local">
29     <option value="0" selected>Selecione</option>
30     <option value="Recife">Recife</option>
31     <option value="São Paulo">São Paulo</option>
32     <option value="Belo Horizonte">Belo Horizonte</option>
33     <option value="Campinas">Campinas</option>
34   </select>
35
36   <br><br>
37   <input type="submit" value="Enviar">
38   <input type="reset" value="Limpar formulário">
39 </form>

```

5

PHP na web



Formulários HTML

- Para pegar os dados vindos da requisição HTTP, no PHP, utilizamos as variáveis superglobais **\$_GET** e **\$_POST**

```

10 <h2>Formulário</h2>
11 <form action="pagina.php" method="post">
12   <label for="nomefuncionario"><b> Nome do funcionário</b></label>
13   <input type="text" name="nomefuncionario" id="nomefuncionario"><br>
14
15   <span><b> Alocação</b></span>
16   <label for="alocadoSim">Sim</label>
17   <input type="radio" name="alocado" id="alocadoSim">
18   <label for="alocadoNao">Não</label>
19   <input type="radio" name="alocado" id="alocadoNao"><br>
20
21   <span><b> Habilidades</b></span>
22   <label for="habilidadePHP">PHP</label>
23   <input type="checkbox" name="habilidades" id="habilidadePHP">
24   <label for="habilidadeJava">Java</label>
25   <input type="checkbox" name="habilidades" id="habilidadeJava"><br>
26
27   <label for="local"><b> Local</b></label>
28   <select name="local" id="local">
29     <option value="0" selected>Selecione</option>
30     <option value="Recife">Recife</option>
31     <option value="São Paulo">São Paulo</option>
32     <option value="Belo Horizonte">Belo Horizonte</option>
33     <option value="Campinas">Campinas</option>
34   </select>
35
36   <br><br>
37   <input type="submit" value="Enviar">
38   <input type="reset" value="Limpar formulário">
39 </form>

```

6

Pilha de execução (*Call Stack*)



- Como em qualquer outra linguagem de programação, o *call stack* (ou "pilha de chamadas") é uma estrutura de dados que registra as chamadas de função ou método em execução na aplicação. Quando uma função ou método é chamado, um novo frame é adicionado à pilha, contendo informações sobre a função ou método chamado, bem como sobre o estado atual da execução da aplicação.
- O *call stack* permite que a aplicação saiba em qual ponto da execução se encontra, o que é importante para garantir que as chamadas de função sejam executadas e encerradas na ordem correta, sem conflitos ou erros.

Copyright© 2023 Accenture All Rights Reserved

7

7

Pilha de execução (*Call Stack*)



- É importante entender a pilha de execução para conseguir identificar erros ocorridos, principalmente quando se trata de projetos grandes
- Existem 3 tipos de erros que podem ocorrer em um programa: de sintaxe, de execução e de semântica
- Para identificar e corrigir erro, devemos depurar o programa

Copyright© 2023 Accenture All Rights Reserved

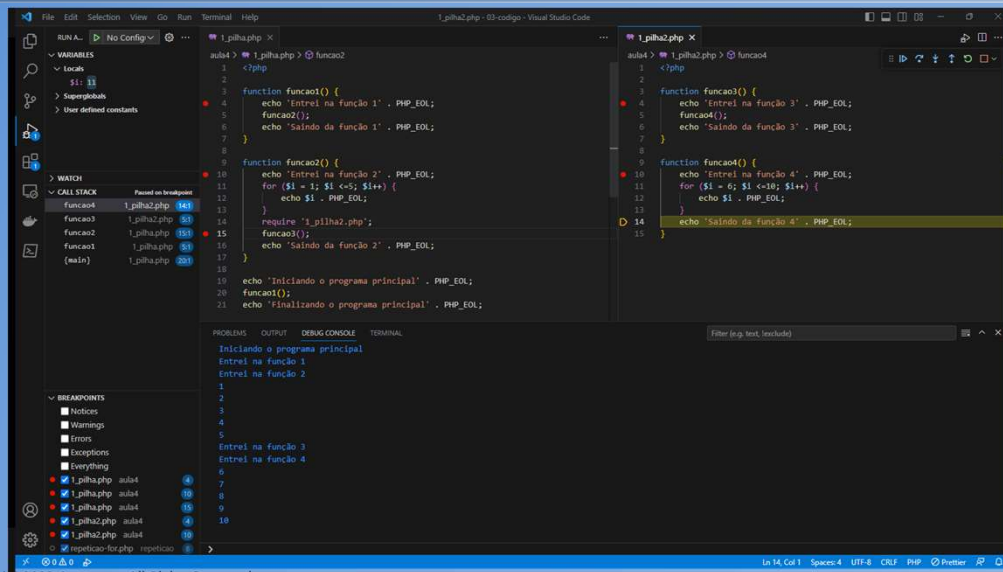
8

8

Pilha de execução (Call Stack)



Debug



Copyright© 2023 Accenture All Rights Reserved

9

Erros e Exceções



- No PHP, **erros** e **exceções** são duas formas de lidar com situações indesejadas ou excepcionais que ocorrem durante a execução de um programa.
- Embora ambas se refiram a problemas no código, elas são tratadas de maneiras diferentes.

Copyright© 2023 Accenture All Rights Reserved

10

10

Erros e Exceções



Erros:

- são problemas que geralmente indicam falhas no funcionamento do PHP ou violações nas regras da linguagem
- interrompem a execução normal do programa
- Exemplos comuns de erros são "*Parse error: syntax error*", "*Fatal error: Call to undefined function*"

Exceções:

- são eventos excepcionais ou situações anormais que ocorrem durante a execução de um programa, mas que podem ser previstas e tratadas
- Podem ser capturadas e tratadas por meio de blocos **try-catch**, permitindo que o programa lide com o problema de forma adequada e continue sua execução normalmente

Copyright© 2023 Accenture All Rights Reserved

11

11

Erros e Exceções



- O PHP retornará erros em resposta a inúmeras condições de erros internos. Pode ser utilizado para sinalizar inúmeras condições diferentes, e podem ser exibidos e/ou registrados em log se necessário.
- Cada erro que o PHP gera inclui um tipo. A lista destes tipos de erros está disponível, junto a uma breve descrição de seu comportamento e como podem ser causados:
https://www.php.net/manual/pt_BR/errorfunc.constants.php
- Se nenhuma manipulação de erro for definida, o PHP tratará todos os que ocorrerem de acordo com sua configuração

Copyright© 2023 Accenture All Rights Reserved

12

12

Erros e Exceções



Arquivo php.ini

```

475 ; E_COMPILE_ERROR|E_RECOVERABLE_ERROR|E_ERROR|E_CORE_ERROR (show only errors)
480 ; Default Value: E_ALL
481 ; Development Value: E_ALL
482 ; Production Value: E_ALL & -E_DEPRECATED & -E_STRICT
483 ; https://php.net/error-reporting
484 error_reporting = E_ALL
485
486 ; This directive controls whether or not and where PHP will output errors,
487 ; notices and warnings too. Error output is very useful during development, but
488 ; it could be very dangerous in production environments. Depending on the code
489 ; which is triggering the error, sensitive information could potentially leak
490 ; out of your application such as database usernames and passwords or worse.
491 ; For production environments, we recommend logging errors rather than
492 ; sending them to STDOUT.
493 ; Possible Values:
494 ;   Off - Do not display any errors
495 ;   stderr = Display errors to STDERR (affects only CGI/CLI binaries!)
496 ;   On or stdout = Display errors to STDOUT
497 ; Default Value: On
498 ; Development Value: On
499 ; Production Value: Off
500 ; https://php.net/display-errors
501 display_errors = On
502
503 ; The display of errors which occur during PHP's startup sequence are handled
504 ; separately from display_errors. We strongly recommend you set this to 'off'
505 ; for production servers to avoid leaking configuration details.
506 ; Default Value: On
507 ; Development Value: On
508 ; Production Value: Off
509 ; https://php.net/display-startup-errors
510 display_startup_errors = On
511
512 ; Besides displaying errors, PHP can also log errors to locations such as a
513 ; server-specific log, STDERR, or a location specified by the error_log
514 ; directive found below. While errors should not be displayed on productions
515 ; servers they should still be monitored and logging is a great way to do that.
516 ; Default Value: Off
517 ; Development Value: On
518 ; Production Value: On
519 ; https://php.net/log-errors
520 log_errors = On
521

```

Dica para localizar o php.ini

```

$ php --ini
Configuration File (php.ini) Path:
Loaded Configuration File:      C:\php\php.ini
Scan for additional .ini files in: (none)
Additional .ini files parsed:    (none)

```

Copyright© 2023 Accenture All Rights Reserved

13

13

Erros e Exceções



Diretivas no php.ini para manipulação de erros no PHP

- **error_reporting** – quais erros são reportados e quais são ignorados
- **display_errors** – controla se o erro será mostrado como parte da saída do script
- **log_errors** – logará qualquer erro para um arquivo ou para o syslog definido na diretiva error_log.
- **error_log** – Nome do arquivo onde os erros de script devem ser registrados

Copyright© 2023 Accenture All Rights Reserved

14

14

Tratamento de exceções



• try-catch | try-catch-finally

```

1  <?php
2
3  error_reporting(E_ALL);
4
5  $dividendo = 10;
6  $divisor = 0;
7
8  try {
9
10     $divisao = $dividendo / $divisor;
11     echo "$dividendo / $divisor = $divisao<br>" ;
12
13 } catch (\Throwable $t) {
14
15     echo 'Erro capturado: ' . $t->getMessage();
16 }
17
18 echo 'Fim do programa.';
19

```

```

1  <?php
2
3  error_reporting(E_ALL);
4
5  $dividendo = 10;
6  $divisor = 0;
7
8  try {
9      // Trecho com possível erro
10     $divisao = $dividendo / $divisor;
11     echo "$dividendo / $divisor = $divisao<br>" ;
12 } catch (\Throwable $t) {
13     // Captura do erro
14     echo 'Erro capturado: ' . $t->getMessage();
15 } finally {
16     // Trecho sempre executado (opcional)
17     // Geralmente usado para fechar conexão com banco,
18     // ou fechar arquivo
19     echo 'mensagem qualquer<br>';
20 }
21
22 echo 'Fim do programa.';

```

Copyright© 2023 Accenture All Rights Reserved

15

15

Lançamento de exceções



• Throw (Throwable)

- Exception
- Error
- LogicException
- DomainException

• Exemplo:

```

if (!validaCPF($cpf) {
    throw new DomainException('CPF inválido');
}

```

Copyright© 2023 Accenture All Rights Reserved

16

16

Atividade



Calculadora WEB

- Transforme a calculadora da primeira atividade em uma Calculadora para web. Você deve ter as funções da calculadora separado do arquivo principal.