|  |  |  |
| --- | --- | --- |
| Elemento a estandarizar | Descripción de estándar | Ejemplo de uso |
| Etiqueta PHP en los archivos/ PHP tag in files | Etiqueta php definida sin cierre | <?php |
| Dependencias o archivos de soporte para acciones/ Dependencies or support files for actions | En la línea siguiente sin espacio después de la etiqueta php | <?php  require\_once(‘route.extension’) |
| Clases/Class | En la línea siguiente dejando una línea de espacio. Usando CamelCase en su variación de letra inicial de cada palabra como mayúscula.  Con un espacio | <?php  Require\_once(‘route.extension’)  class EmployeeQueries  {  Class content  } |
| Methods/Metodos | camelCase en su variación de letra inicial de un nombre compuesto por dos palabras minúscula y la siguiente palabra con mayúscula | public function readOne()  {  } |
| Variables de sentencias SQL/ SQL Sentences variables | Definición de la variable que almacena la sentencia de SQL | $query |
| Estructura de sentencia SQL/ SQL sentence structure | Palabras reservadas de forma descendente y palabras de lógica en la consulta en la misma línea | public function searchRows($value)  {  $query = 'SELECT \*  FROM empleados  WHERE nombre\_empleado LIKE = ?  ORDER BY id\_empleado';  $params = array("%$value%")  return Database::getRows($query, $params);  } |
| Variable almacenadora de parámetros/ Params variable | Definicion de la variable que almacena los parámetros de consulta referentes a $query | $params |

Archivos en dao:

Archivos en dto:

|  |  |  |
| --- | --- | --- |
| Elemento a estandarizar | Descripción de estándar | Ejemplo de uso |
| Clases que heredan otras clases/ Classes that extends from other classes | Clases que heredan a otras clases para acceder a sus métodos serán escritas como un objeto que hace referencia a la información que maneja | class Employee extends EmployeeQueries |
| Definición de variables almacenadoras de información para getters y setters/ Definition of storage variables for getters and setters | Palabra que hace referencia a la información que se maneja seguida de un guion bajo y el campo que almacena, todo en minúscula. | protected $employee\_id = null;  protected $employee\_name = null;  protected $employee\_lastname = null;  protected $employee\_dui = null; |
| Metodos set/set Methods | camelCase en su variación de letra inicial de un nombre compuesto por dos o más palabras, con la primera palabra en minúsculas y las siguientes palabras con la primera letra mayúscula | Public function setEmployeeDate($value)  {  } |
| Metodos get/get Methods | camelCase en su variación de letra inicial de un nombre compuesto por dos o más palabras, con la primera palabra en minúsculas y las siguientes palabras con la primera letra mayúscula | public function getEmployeeDate()  {  return $this->employee\_date;  } |

Archivos en bussiness:

|  |  |  |
| --- | --- | --- |
| Elemento a estandarizar | Descripción de estándar | Ejemplo de uso |
| Objeto de clases externas / External class objects | Serán escritas como un objeto que hace referencia a la información que maneja seguido de un guion bajo y la palabra model. | $employee\_model = new Employee; |
| Casos en las acciones / Cases in actions | El nombre de los casos será escrito con el mismo nombre que se le asigno en el método programado en el modelo. | switch ($\_GET['action']) {  case 'readAll':  if ($result['dataset'] = $employee\_model->readAll()) {  $result['status'] = 1;  $result['message'] = 'Data was found';  } elseif (Database::getException()) {  $result['exception'] = Database::getException();  } else {  $result['exception'] = 'No data to show';  }  break; |
| Mensajes de respuesta / Response messages | Todos los mensajes de respuesta referentes a las acciones definidas en el switch serán escritos en el idioma inglés. | $result['message'] = 'Data was found'; |
| Else if / elseif | En el proyecto usaremos el else if, de forma unida | If()  {  }elseif()  {  } |

Archivos en JavaScript:

|  |  |  |
| --- | --- | --- |
| Elemento a estandarizar | Descripción de estándar | Ejemplo de uso |
| Constantes/ Const | Todas las constantes en los archivos de JavaScript serán definidas con mayúsculas y en inglés. | const EMPLOYEE\_API = 'bussines/dashboard/employee.php';  const SAVE\_FORM = document.getElementById('save-form'); |
| Métodos de procesos SCRUD/SCRUD process methods | Todos los métodos referentes a los procesos de un SCRUD serán definidos bajo el estilo CamelCase en su variante de la primera palabra del nombre en minúscula y la siguiente en mayúscula, esto en caso de estar compuesto por dos palabras, de lo contrario, solo la primera palabra seria minúscula, las siguientes serian mayúscula | function createEmployee() {  MODAL\_TITLE.textContent = 'CREATE EMPLOYEE';  fillSelect(EMPLOYEE\_API, 'readTypes', 'types');  document.getElementById('update').style.display = 'none';  document.getElementById('clean').style.display = 'block';  } |
| Métodos personalizados/ Customize methods | Todos los métodos que sean propios de un funcionamiento diferente a los procesos SCRUD, deberán tener un nombre que haga referencia a su función y deberá ser escrito con letra inicial mayúscula. | const Clean = () => {  document.getElementById('id').value = '';  document.getElementById('Employee\_name').value = '';  document.getElementById('Employee\_lastname').value = '';  document.getElementById('Employee\_dui').value = '';  document.getElementById('Employee\_phone').value = '';  document.getElementById('Employee\_email').value = '';  document.getElementById('Employee\_date').value = '';  document.getElementById('Employee\_address').value = '';  } |

Archivos en HTML

|  |  |  |
| --- | --- | --- |
| Elemento a estandarizar | Descripción de estándar | Ejemplo de uso |
| Título de página/ Page Title | El título de la página deberá ser solamente un nombre que haga referencia a la información que se muestre en la interfaz activa, siempre en inglés y en plural. | <title>Employees</title> |
| Id de inputs / Inputs Ids | El nombre de los identificadores en los campos deberá iniciar con mayúscula, de estar compuesto por dos o mas palabras, cada palabra distinta deberá ir separada por un guion bajo, y la palabra que siga al guion debe ser minúscula. | <input type="text" required id="Employee\_name" /> |
| Nombre de inputs / Inputs name | El nombre de los campos deberá iniciar con minúscula, de estar compuesto por dos o más palabras, cada palabra distinta deberá ir separada por un guion bajo, y la palabra que siga al guion debe ser minúscula. | <input type="text" required name="employee\_name" /> |

Generalidades:

El nombramiento de los archivos y directorios sin importar su extensión debe realizarse en minúscula.

* Dao:

Los archivos en el directorio deberán estar escritos con un nombre que haga referencia a la información que maneja el archivo en singular seguida de un guion bajo y la palabra “queries” siempre en minúscula.

* Dto:

Los archivos en el directorio deberán estar escritos con un nombre que haga referencia a la información que maneja el archivo en singular.

* Bussiness:

Los archivos en el directorio deberán estar escritos con un nombre que haga referencia a la información que maneja el archivo en singular.

* JavaScript:

Los archivos en el directorio deberán estar escritos con un nombre que haga referencia a la información que maneja el archivo en singular a excepción de “components.js”.

* HTML:

Los archivos en el directorio deberán estar escritos con un nombre que haga referencia a la información que maneja el archivo en singular a excepción de “components.js”.

El idioma definido para todos los archivos de HTML es inglés.

Todos los mensajes de respuesta en acciones deberán ser escritos en inglés.

Los títulos de alertas y mensajes asociados deben ser escritos en inglés.