Avançando com Orientação a Objetos com PHP: Herança, Polimorfismo e Interfaces

- Esse curso capacita a:
    - Configurar seus namespaces;
    - Utilizar “alias” para classes com mesmo nome;
    - Utilizar corretamente uma interface com PHP;
    - Entender a diferença entre classes abstratas e interfaces;
    - Como utilizar herança e classes abstratas;
    - Como garantir que um método não seja sobrescrito;

Aulas:

1. Herança:
    - Herança é o terceiro pilar da orientação a objetos;
    - Para acessar um membro da classe pai, na qual está sendo herdada -> “parent”;
    - PHP apresenta modificador de acesso “protected”;

2. Namespace e Autoload:
    - Deve-se separar as classes em namespaces, assim como os arquivos em pastas, com isso é possível ter classes com o mesmo nome em namespaces diferentes;
    - Utilizando um namespace raiz evita-se conflitos com classes de pacotes externos que possam ser utilizados;
    - Para utilizar uma classe, é necessário o “use” ou informar seu nome completo;
    - Para fazer um Autoloader com PHP utiliza-se a função “spl_autoload_register”, com isso evita-se ficar utilizando o “require” para incluir os arquivos necessários a execução;

3. Classes e métodos abstratos:
    - Classes abstratas: são classes que ainda não estão prontas para serem instanciadas e precisam ser estendidas;
    - Métodos abstratos são uma forma de obrigar classes filhas a implementar determinado método;
    - Apenas classes abstratas podem ter métodos abstratos;

4. Polimorfismos:
    - Polimorfismo é a capacidade de um objeto poder ser referenciado ou se comportar de várias formas dependendo do contexto;
    - Classes de serviço, são classes que representam uma funcionalidade ao invés de representar um modelo do domínio;
    - 4° Pilar da Orientação a Objetos é o Polimorfismo;

5. Interfaces:
    - Herança múltipla não é permitido em PHP. Uma forma de superar essa ausência é utilizando interfaces, que são classes abstratas que contém apenas métodos abstratos;
6. Métodos mágicos:
    - São métodos chamados automaticamente em determinados momentos;
    - Quando uma classe é criado, ela apresenta métodos próprios que já vieram prontas pelo PHP, mas é possível implementa-las, por exemplo, (__toString) e (__get);
    - Traits são uma forma de reutilizar código, sem necessidade de herança, é como um “use”, só que ele vem dentro da classe, e não entre o nome da classe e o namespace;