Properties Overloading কি?

এটি একটি খুব গুরুত্বপূর্ন প্রশ্ন যে Properties Overloading কি? হ্যা আমরা এটি জানবো তবে তার আগে জেনে নেই বাস্তবিক ক্ষেত্রে আমরা overloading বলতে কি বুঝি। ধরুন আপনি আপনার বড় ভাইয়ের বিয়েতে গেছেন, সেখানে আপনার বেয়াইনেরা ধরেছে মিষ্টি খাওয়ার জন্য (খুব খুশি হয়ে গেলেন মনে হচ্ছে? এটি নিছকই একটা কল্পনা মাত্র :D ) আপনি মিষ্টি খাওয়া শুরু করলেন। স্বাভাবিকভাবেই আপনি ৫-৬ টি মিষ্টি খেতে পারেন কিন্তু আপনার বেয়াইনগুলো নাছড়বান্দা তারা আপনাকে ১৫টা মিষ্টি না খাইয়ে ছাড়বেই না, খেতেই হবে। তখন এ মিষ্টিগুলি আপনার জন্য কি হবে? হ্যা ঠিক ধরেছেন, এই মিষ্টিগুলো তখন আপনার জন্য overload হয়ে যাবে কারন এটি আপনার সমর্থের বাহিরে অর্থাৎ আপনার পেটে ৫-৬ টির বেশি এক্সেস নেই। যেহেতু এক্সেস নেই তার তারপরেও আপনাকে খেতে হবে তাহলে যে ঘটনাটি দারালো তাহলো Misty Overloading.

অনেক তো মিষ্টি খেলেন কিন্তু Properties Overloading তো বোঝা হলো না। হুম এব্যপারটিও ঠিক মিষ্টি খাওয়ার মতই। ধরুন আপনি একটি class তৈরী করলেন যেখানে $kola নামে কোনো properties ই নেই কিন্তু আপনি বাহির থেকে $kola প্রোপার্টিজকে কল করলেন অথবা আপনি বাহির থেকে $kola প্রোপার্টিজ কে কোনো ভ্যালু এসাইন করলেন এ ক্ষেত্রে যে ঘটনাটি ঘটে এটিই হলো Properties Overloading. আমরা একটা উধাহরন দিয়ে দেখিঃ

**class** **Fruits**

{

}

$fruits = **new** Fruits;

**echo** $fruits->kola;

দেখুন আমাদের Fruits class এ $kola নামে কোনো প্রোপার্টিজ নেই কিন্তু আমি সেটিকে কল করেছি যার ফলে নিচের error টি দেখাবে

Properties Notfound

এবার বোঝা গেলো Properties Overloading কি? বুঝলে মিষ্টি খাওয়ান।

তাহলে কিভাবে আমরা এই সমস্যাটিকে হ্যান্ডল করবো? হা ভাই PHP Magic Method এসকল কিছুই খুব স্মার্টলি হ্যান্ডল করবে \_\_set() ও \_\_get()method এর মাধ্যমে। এদুটি Magic Method হলো PHP’র বহুল ব্যবহৃত দুটি মেথড। তো চলুন আজকে দেখে নেই কিভাবে এ Magic Method দুটি কাজ করে।

Assign Value With \_\_set()

এ Magic Method টির দুটি প্যারামিটার \_\_set($name, $value) . $name প্যারামিটারটি মূলত Properties এর নামকে mention করে আর $value প্যারামিটারটি এর ভ্যালুকে মেনশন করে। তাহলে \_\_set() কি কাজ করবে? হ্যা, আপনি যখন কোনো class এর Inaccessible Properties কে কোনো ভ্যালু এসাইন করবেন ঠিক তখনই এ মেথডটি ট্রিগার হবে অর্থাৎ এটি সয়ংক্রিয়ভাবে এক্সিকিউট করবে। কি, মজার না? চলুন একটা উধাহরন দিয়ে দেখিঃ

**class** **Fruits**

{

**public** **function** **\_\_set**($name, $value)

{

**echo** $name. ' = '. $value;

}

}

$fruits = **new** Fruits;

$fruits->kola='Sagor Kola';

**Output:**

kola = Sagor Kola

খেয়াল করুন আমরা এখানে $kola Properties টিকে একটি ভ্যালু এসাইন করেছি কিন্তু বাস্তবে এর কোনো অস্তিত্বই নেই, PHP যখনই এরকম কোনো ইভেন্ট পায় সাথে সাথে এর অভ্যন্তরিন যদি \_\_set() method টি থাকো তবে সেটিকে এক্সিকিউট করে। আর এখানে ঠিক সেরকমই ঘটেছে। আমরা কিন্তু বাহির থেকে এই মেথডটি কে কল করিনি (অবশ্য করাও যাবে না) তারপরেও যখন একটি absent প্রোপার্টিজ কে কোনো ভ্যালু এসাইন করেছি ঠিক তখনই \_\_set() method টি এক্সিকিউট হয়েছে। কি মনে হচ্ছে, অনেক কঠিন? আরেকটু অপেক্ষা করুন আরো মজার ব্যপার আছে।

Accessing Value With \_\_get()

এ Magic Method টির একটি প্যারামিটার \_\_get($name) . $name যেটি প্রোপার্টিজের নামকে মেনশন করে। অর্থাৎ আমি যে Inaccessible প্রোপার্টিজটিকে খুজছি এটা তারই নাম। যখন কেউ class এর কোনো Inaccessible properties কে কল করে ঠিক তখন এই মেথডটি সয়ংক্রিয়ভাবে ট্রিগার হয়। একটি উধাহরন দেখিঃ

**class** **Fruits**

{

**public** **function** **\_\_set**($name, $value)

{

**echo** $name. ' = '. $value;

}

**public** **function** **\_\_get**($name)

{

**echo** "You call : " . $name;

}

}

$fruits = **new** Fruits;

**echo** $fruits->alu;

**Output:**

You call : alu

দেখুন আমাদের $alu নামে কোনো প্রোপার্টিজ নেই, যেহেতু এর এই নামে কোনো প্রোপার্টিজ নেই কিন্তু আমি একে কল করেছি তাই \_\_get() মেথডটি এক্সিকিউট হয়েছে। অর্থাৎ PHP যখনই এরকমের কোনো ইভেন্ট পায় ঠিক তখনই \_\_get() মেথডটি ট্রিগার করে।

চলুন একটা বাস্তবিক উধাহরন দিয়ে শেষ করি। আমরা \_\_set() ও\_\_get() মেথড ব্যবহার করে Laravel এর Eloquent ORM এর মতো একটি Insert Query অপারেশন তৈরী করবো যেটি যেকোনো নামের প্রোপার্টিজে ভ্যালু এসাইন করে এরপর insert() method কে কল করলে উক্ত প্রোপার্টিজ ও ভ্যালুগুলো নিয়ে একটি INSERT query তৈরী করে স্ট্রিং আকারে রিটার্ন করবে।

**class** **ORM**

{

**protected** $\_data = [];

**public** **function** **\_\_set**($name, $value)

{

**if**(is\_array($this->\_data)) {

$this->\_data[$name] = $value;

}

}

**public** **function** **\_\_get**($name)

{

**return** $this->\_data[$name];

}

**public** **function** **insert**($table)

{

$key = array\_keys($this->\_data);

$val = array\_values($this->\_data);

$fields = implode(', ', $key);

$value = array\_map([$this, 'makeSqlString'], $val);

$value = implode(', ', $value);

**return** 'INSERT INTO '. $table.'('.$fields.') VALUES('.$value.')';

}

**protected** **function** **makeSqlString**($val)

{

$types = ['integer', 'double', 'float'];

**if**(in\_array(gettype($val), $types)) {

**return** $val;

}

**return** "'{$val}'";

}

}

$db = **new** ORM;

$db->name = 'The Alien';

$db->type = 'Anonymous';

$db->age=800;

**echo** $db->insert('tbl\_users');

Output:

INSERT INTO tbl\_users(name, type, age) VALUES('The Alien', 'Anonymous', 800)

দেখুন এখানে আপনি চাই‍লে যেকোনো নামে প্রোপার্টিজে ভ্যালু পাঠাতে পারবেন যেটি \_\_set() একটি array তে স্টোর করে এবং সেই এ্যারেটি ব্যবহার করে একটি INSERT query ‘insert()` মেথডের মাধ্যমে তৈরী করে যেটি একটি টেবিলের নাম গ্রহন করে।

আর এভাবেই Properties Overloading এর মাধ্যমে আপনি চমৎকার ও ইফেক্টিভ কাজ করতে পারেন। আশা করি মজা পেয়েছেন :)