**JS. ООП**

## Основы ООП

* **Инкапсуляция.** Инкапсуляция - это свойство системы, позволяющее объединить данные и методы, работающие с ними, в классе и скрыть детали реализации от пользователя.

Инкапсуляцию можно сравнить с работой автомобиля с точки зрения типичного водителя. Многие водители не разбираются в подробностях внутреннего устройства машины, но при этом управляют ею именно так, как было задумано. Пусть они не знают, как устроен двигатель, тормоз или рулевое управление, — существует специальный интерфейс, который автоматизирует и упрощает эти сложные операции. Сказанное также относится к инкапсуляции и ООП — многие подробности "внутреннего устройства" скрываются от пользователя, что позволяет ему сосредоточиться на решении конкретных задач. В ООП эта возможность обеспечивается [классами](http://www.php.su/learnphp/phpoo/?classes), [объектами](http://www.php.su/learnphp/phpoo/?classes) и различными средствами выражения иерархических связей между ними.

* **Полиморфизм.** Полиморфизм позволяет использовать одни и те же имена для похожих, но технически разных задач. Главным в полиморфизме является то, что он позволяет манипулировать объектами путем создания стандартных интерфейсов для схожих действий. Полиморфизм значительно облегчает написание сложных программ.
* **Наследование.** Наследование позволяет одному объекту приобретать свойства другого объекта, не путайте с копированием объектов. При копировании создается точная копия объекта, а при наследовании точная копия дополняется уникальными свойствами, которые характерны только для производного объекта.

## Класс

**Класс** - это базовое понятие в объектно-ориентированном программировании (ООП). Экземпляр класса - это **объект**. Свойства и методы называются членами класса. Вообще, объектом является все то, что поддерживает инкапсуляцию. Если класс можно рассматривать как тип данных, то объект — как переменную (по аналогии). По общепринятым правилам имена классов ООП начинаются с прописной буквы.

Базовый синтаксис выглядит так:

class MyClass {

// методы класса

constructor() { ... }

method1() { ... }

method2() { ... }

method3() { ... }

...

}

Методы в классе не разделяются запятой! Затем используйте вызов new MyClass() для создания нового объекта со всеми перечисленными методами. При этом автоматически вызывается метод constructor(), в нём мы можем инициализировать объект. Например:

class User {

constructor(name) {

this.name = name;

}

sayHi() {

alert(this.name);

}

}

// Использование:

let user = new User("Иван");

user.sayHi();

lkj

class Имя\_класса {

// описание членов класса - свойств и методов для их обработки

}

**Объект** = new Имя\_класса;