УДК 004.4

СРАВНИТЕЛЬНЫЙ АНАЛИЗ РЕАЛИЗАЦИИ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА НА ЯЗЫКАХ JAVASCRIPT И PHP

Мавлянов А. Н.
Университет ИТМО

В данной работе отражён сравнительный анализ объектно-ориентированного подхода на языках JavaScript и PHP. Критериями сравнительного анализа являются частота использования объектно-ориентированных конструкций и модель реализации объектно-ориентированного подхода. В результате сравнительного анализа были выявлены как аналогичные, так и противоположные особенности реализации объектно-ориентированного подхода на рассматриваемых языках программирования.

На настоящем этапе высокие требования предъявляются к внутренней языковой мобильности разработчиков в составе фулл-стэк команд, что обусловливает актуальность данной работы, посвящённой объединяющим и различающим чертам имплементации объектно-ориентированного подхода в JavaScript и PHP.

Целью настоящей работы является сравнительный анализ реализации объектно-ориентированного подхода к программированию на языках JavaScript и PHP, позволяющий разработчикам свободнее ориентироваться в кодовой базе при переходе от одной экосистемы к другой.

Выделим модели реализации объектно-ориентированного подхода, на основе которых можно провести сравнительный анализ относительно PHP и JavaScript:

1. Прототипная модель
2. Модель на основе классов

ООП на языке JavaScript реализуется с помощью прототипной модели, которая не содержит как таковые классы, а имитирует поведение класса, расширяя существующие объекты (прототипы). При имплементации ООП на языке PHP используется модель, основанная на классах. Кроме этого, для реализации ООП-подхода на языке PHP используются аспекты основных парадигм ООП:

1. Наследование
2. Инкапсуляция
3. Полиморфизм

Наследование на языке PHP реализуется на основе родительского класса, от которого наследуются и другие дочерние классы, то время как на языке JavaScript применяется наследование от конкретного объекта к конкретному объекту. В современных версиях JavaScript наследование может быть реализовано с помощью методов Object.create и Object.setPrototypeOf.

Инкапсуляция на языке PHP предполагает функционал ограничения доступа к данным и реализуется с помощью таких служебных слов как «public», «protect», «private». Данные служебные слова определяют уровень защищенности данных от внешнего воздействия. В отличие от PHP, на языке JavaScript подобный эффект достигается за счёт функции конструктора, который скрывает свою реализацию. Это позволяет ограничивать доступ к данным, которые находятся внутри тела данной функции. При этом на стадии обсуждения находятся изменения стандарта, легализующие указанные служебные слова с учётом принятой в JavaScript прототипной модели.

Следующим критерием, на основе которого можно провести сравнительный анализ реализации объектно-ориентрованного подхода, является частота использования объектно-ориентированных конструкций. Для этого рассматриваются популярные фреймворки и библиотеки JavaScript и PHP.

В современной экосистеме представлены такие JavaScript-библиотеки и фреймворки как React.js, Angular.js, Vue.js, Ember.js, Polymer, Aurelia, Sail.js которые реализуют объектно-ориентированные конструкции. Наряду с этими инструментами существует и такие фреймворки как Express.js, Koa.js, Hapi.js, Данные фреймворки не реализует объектно-ориентированный подход, а реализует подход промежуточного программного обеспечения (middleware). Данный подход имплементируется на основе принципов функционального программирования, где используются чистые функции и функции высшего порядка, а не классы или экземпляры (объекты) данных классов.

Практически все PHP-фреймворки являются объектно-ориентированными, в отличие от JavaScript фреймворков. Например, такие популярные PHP-фреймворки как Laravel, Yii, Symfony, Phalcon, CodeIgniter, CakePHP, Zend, FuelPHP, PHPixie, Slim основываются на принципах объектно-ориентированного подхода к программированию при решении тех или иных предметных задач.

Будучи двумя основными языками веб-разработки, JavaScript и PHP развиваются конвергентно, дополняя друг друга. PHP тяготеет к приоритету объектно-ориентированного подхода, но по мере развития обогащается функциональными чертами. Например, в PHP 7.4 введены короткие замыкания, также известные как стрелочные функции. С другой стороны, в JavaScript версии 6 (ES6) ключевое слово class стало так называемым синтаксическим сахаром для создания множества объектов по прототипу.

В результате анализа можно заключить, что в экосистеме JavaScript существует тенденция к постепенному росту доли использования объектно-ориентированных конструкций и паттернов при одновременном развитии существующих функциональных аналогов, в то время как в PHP наблюдается внедрение в язык типичных для JavaScript синтаксических элементов типа стрелочных функций при изначально обширном объектно-ориентированном фундаменте.