РЕАЛИЗАЦИЯ СИСТЕМЫ СЕРИАЛИЗАЦИИ И ДЕСЕРИАЛИЗАЦИИ МОДЕЛЕЙ В OBJECTIVE-C

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Тускенис Д. С., Рогов М. Г.

Фролов И. И. – кандидат технических наук

При разработке пользовательских приложений очень часто имеет место задача преобразования моделей объектов из формата обмена данными, таких как JSON или XML, в экземпляры классов на конкретном языке программирования. В отличие от Java, в языке Objective-C не предусмотрено стандартных средств для такого преобразования.

Так как в стандартной библиотеке языка Objective-C присутствуют средства для сериализации и десериализации Foundation-объектов (строки, массивы, словари), задача сводится к разработке механизма отображения объектов произвольных классов в объекты стандартных классов и наоборот. Обычно объект какого-то класса отображается в виде словаря, в котором ключом является название свойства объекта, а значением – значение свойства. Если объект имеет свойства составного типа (объект внутри объекта), то значением в этом случае будет еще один словарь.

Несмотря на наличие достаточно мощного механизма интроспекции, реализация системы сериализации в Objective-C является нетривиальной задачей (как, например, в Java). Рефлексия в Objective-C представлена специальной библиотекой, которая определяет функции, с помощью которых можно узнать всю информацию о классе на этапе выполнения программы. Кроме того, для работы со значениями свойств используется методология KVC, которая определяет интерфейсы для получения доступа к свойствам по ключам (строкам).

Возможности разработанной системы сериализации включают:

1) возможность отображения свойств любого типа (стандартного класса, произвольного класса, примитивного типа);
2) наличие механизма «отображения ключей», когда название свойства исходного и преобразованного объекта не совпадают;
3) способ указания опциональных свойств (значение которых может отсутствовать) и свойств, которые следует игнорировать при отображении;
4) рекурсивное преобразование свойств-агрегатов (например, массивов);
5) наличие специального протокола, с помощью которого можно настроить систему так, чтобы определенные свойства или свойства определенного типа отображались не стандартным способом. Примером ситуации, где такая функция может понадобиться, может служить формат представления даты и времени.

Преобразование произвольного объекта в словарь состоит в том, чтобы получить список свойств объекта и сформировать пары вида «ключ-значение», где ключом будет название свойства (либо его псевдоним), а значением – объект, оборачивающий значение свойства (например, строка или словарь). В результате получаем словарь, который можно сериализовать стандартными средствами.

Обратная задача является более сложной, так как в данном случае нужно описать то, что мы хотим получить из входного объекта, то есть, объект какого класса вернет функция преобразования. Минимальный объем информации, необходимый для такого преобразования, включает в себя лишь класс выходного объекта.

Однако для свойств, значением которых является массив объектов, этой информации недостаточно, так как в Objective-C для массивов нельзя указать тип объектов, содержащихся в нем. На практике можно пренебречь тем фактом, что массив может содержать объекты разных классов, так как чаще всего модель данных не использует такую возможность. Тогда для указания типа объектов в массиве можно воспользоваться специальным синтаксисом. Вместо указания типа свойства как массива можно указать его как массив, поддерживающий определенный протокол. Указанный протокол имеет то же название, что и класс объектов в данном массиве. Такой протокол не имеет методов, но позволяет в ходе интроспекции получить информацию о том, что свойство имеет тип массива, поддерживающего этот протокол, и при десериализации будет известно, во что преобразовывать объекты в массиве.

Таким образом, разработанная система сериализации избавляет разработчика от необходимости написания специализированных алгоритмов преобразования моделей. Использование системы в общем случае не требует модификации заголовков модели данных. Дополнительно усовершенствовать систему можно, добавив возможность выносить свойства вложенных объектов на уровень выше или наоборот, «упаковывать» свойства.

Список использованных источников:

1. Kochan S. G. Programming in Objective-C 2.0 / S. G. Kochan // Developer's Library, Addison-Wesley Professional. – Upper Saddle River, NJ, 2011.
2. Lee, K. Pro Objective-C / K. Lee. – 233 Spring Street, New York, NY, 2013.
3. Fairbairn, C. Objective-C Fundamentals / C. Fairbairn, J. Fahrenkrug, C. Ruffenach. – 20 Baldwin Road, Shelter Island, NY, 2012.