Краткое содержание

Часть IV. Связи ресурсов

Часть V. Коллективные операции

Часть VI. Безопасность

Оглавление

Часть II. Принципы проектирования

Часть III. Основы

Часть IV. Связи ресурсов

Часть V. Коллективные операции

Часть VI. Безопасность

Посвящается Ка-эль и Луке. Вы потрясающие.

Предисловие

Все началось с электронной ударной установки. Летом 2019 года один мой друг увлек меня игрой на ней, и я погрузился в это занятие с головой. Иногда я действительно играл на барабанах, но бóльшую часть времени все же проводил за написанием кода, который позволял управлять конфигурацией установки с помощью команд MIDI SysEx.

Когда нагрянула пандемия COVID-19, у меня внезапно появились иные приоритеты, направленные на настройку аудиовизуальной связи в нашей местной церкви, где мы перешли в формат удаленного проведения служб и рассматривали возможности возобновления личных встреч. Для этого мне пришлось изучать протоколы VISCA, NDI и OSC (для камер и аудиомикшеров), а также осваивать программно-ориентированную интеграцию с Zoom, VLC, PowerPoint, Stream Deck и др.

Эти проекты не несут в себе больших объемов бизнес-логики. Практически весь код относится к интеграции, что одновременно и раздражает, и дает большие возможности. Раздражение вызывали протоколы, которые либо были плохо задокументированы, либо не предназначались для тех задач, которые я пытался реализовать с их помощью, либо просто не согласовывались друг с другом. Положительная сторона всего этого заключается в том, что, как только вам удастся разобраться с интеграцией, вы сможете очень легко писать полезные приложения.

Несмотря на то что моя работа в последние несколько лет заключалась в основном в локальной интеграции, точно такие же раздражение и ощущение больших

возможностей возникают и при работе с веб-API. Каждый опыт выбора нового веб-API представляет собой кривую, состоящую из эмоциональных реакций в виде волнения, замешательства, раздражения, принятия и достигаемого в итоге умиротворения. Как только ты досконально осваиваешь мощный API, возникает чувство, что ты являешься дирижером величественного оркестра, готового играть любую предложенную тобой музыку, — даже если ноты скрипачей согласуются лишь под конец, а для группы медных духовых без каких-либо очевидных причин приходится использовать палочку другого цвета.

Данная книга сама по себе не изменит положение дел. Это всего лишь книга. Но если после прочтения вы будете следовать указаниям, изложенным в ней, то она поможет вам обеспечить пользователям более качественный опыт. Если же эту книгу прочтет множество людей, то вместе мы сможем добиться сдвига в направлении более согласованного и менее раздражающего опыта работы с API.

Важно понять, что сама эта книга в целом гораздо более ценна, чем сумма ее фрагментов. Относительно любого из разбираемых Джей Джеем паттернов любая команда может сделать свой обдуманный выбор (хотя кто-то может упустить некоторые из указанных здесь пограничных случаев). Для конкретной ситуации этот выбор может оказаться даже более удачным, чем предложенные в книге рекомендации, ввиду ограниченных требований контекста. Такой подход позволяет получить множество локальных оптимальных решений, но дает очень фрагментированную общую картину, поскольку в этом случае даже в API одной компании потенциально применяется несколько разрозненных техник.

Помимо согласованности для любой отдельно взятой задачи, книга также предлагает единый подход, охватывающий множество сфер проектирования API. Разработчикам программных интерфейсов редко предоставляется пространство для углубленного анализа этих аспектов, и я считаю, что мне очень повезло работать с Джей Джеем и другими (в частности, с Люком Снирингером (Luke Sneeringer) над обсуждением многих тем книги. Я счастлив, что вклад, внесенный компанией Google в проектирование API, может быть весьма полезным многим разработчикам благодаря этой книге и системе AIP, расположенной по адресу https://aip.dev.

Я абсолютно уверен в ценности этой книги, но она не упростит разработку API. Однако она позволит вам избавиться от ненужных сложностей, которые возникают в ходе проектирования интерфейсов, давая возможность сосредоточиться только на том, что действительно уникально для ваших API. Будьте готовы думать, и думать усиленно, но также знайте, что результатом этих размышлений может стать API, работа с которым доставит лишь удовольствие. Ваши пользователи могут никогда не поблагодарить вас за это открыто. Ведь

хорошо спроектированный API зачастую воспринимается как само собой разумеющийся, хоть и является результатом огромных усилий. Но зато вы сможете спать спокойно, зная, что ваши пользователи не будут злиться при работе с API, который пусть и функционирует, но сделан на троечку.

Используйте эту книгу как основу, чтобы построить такой API, который сможет стать фундаментом для других.

Джон Скит (Jon Skeet),
Staff Developer Relations Engineer, Google

Вступление

Многие из нас изучали информатику. Мы анализировали среду выполнения и пространственную сложность алгоритмов с помощью нотации «О большое», знакомились со всевозможными алгоритмами сортировки и изучали различные способы обхода бинарных деревьев. Поэтому, окончив колледж, я планировал, что моя работа будет в первую очередь связана с математикой и точными науками. Но каким же было мое удивление, когда в действительности все оказалось совсем не так.

На деле бóльшая часть работы была связана с проектированием, структурированием и дизайном, а не математикой и алгоритмами. Мне никогда не приходилось думать о том, какой алгоритм сортировки использовать, поскольку для этого всегда имелась библиотека (обычно что-то вроде `array.sort()`). Однако мне все же доводилось долго и упорно размышлять над создаваемыми классами, над функциями для них и параметрами, которые должна принимать каждая функция. Все это оказалось намного более сложным, чем я ожидал.

В реальном мире я узнал, что идеально оптимизированный код далеко не столь же ценен, как хорошо спроектированный. И это оказалось вдвойне верным в отношении веб-API, поскольку они, как правило, имеют гораздо более обширную аудиторию и намного больший спектр случаев применения.

Но здесь напрашиваются вопросы: что значит «хорошо спроектированное ПО»? Что такое «хорошо спроектированный API»? Чтобы ответить на эти вопросы, довольно долго мне приходилось полагаться в основном на случайный набор ресурсов. Для одних тем я отыскивал интересные посты, которые раскрывали популярные современные альтернативы. Для других находил отдельные полезные ответы на Stack Overflow, которые подсказывали нужное направление.

Однако во многих случаях было довольно мало материалов по рассматриваемой теме, и мне приходилось самостоятельно придумывать ответы, надеясь, что мои коллеги не слишком их осудят.

Спустя многие годы подобных изысканий (и таскания повсюду блокнота с надписью «Пугающие проблемы API») я наконец-то решил, что пришло время записать всю информацию, которую я собрал и использовал в работе. Поначалу все это выглядело как набор правил для Google, которые мы с Люком Снирингером систематизировали и которые в конечном итоге стали AIP.dev. Но эти правила выглядели как свод законов. В них говорилось, что нужно делать, но не пояснялось, почему нужно делать именно так. После долгого анализа и исследований, сопровождавшихся непрестанным повторением в уме этого вопроса, я готов представить вам эти правила, а также пояснить их причины.

Несомненно, было бы замечательно, если бы эта книга стала абсолютным решением для всех проблем в мире проектирования API, но, к сожалению, этого не случится. И причина тому проста: как и в архитектуре, выбор любой модели проектирования обычно опирается на личное мнение. Это означает, что одни из вас могут посчитать мои указания удачными и элегантными и будут использовать их для всех последующих проектов. Другие же могут решить, что эта книга демонстрирует отвратительные и чрезмерно ограниченные модели дизайна, и в связи с этим, наоборот, воспринять их в качестве примера того, как не нужно проектировать API. Поскольку я не могу угодить всем, моей единственной целью было предоставить набор проверенных в деле рекомендаций, подкрепив их логическими обоснованиями, объясняющими, почему они выглядят именно так.

Будете ли вы использовать их в качестве примеров для подражания или избегать — дело ваше. По меньшей мере надеюсь, что рассмотренные в книге темы вызовут много дискуссий и инициируют дальнейшую масштабную работу в этом удивительном, сложном и запутанном мире проектирования API.