Пользователь синхронизирует список избранного на телефоне и смарт-часах при помощи интерфейса Bluetooth, далее пользователь добавляет виджет в список активных. После, при нажатии на виджет, происходит инициализация, которая подразумевает выбор пользователем расписания движения общественного транспорта через остановку из списка избранного. В результате данных действий на экране виджета будет отображаться ближайшее время отправление транспорта с остановки по маршруту.

Таким образом, данный сценарий использования позволит ещё быстрее получать доступ к наиболее необходимой информации касательно расписаний движения общественного транспорта, что выгодно отличает данное приложение от аналогов.

Список использованных источников:

1. Документация по Xamarin [Электронный ресурс]. – Электронные данные. – Режим доступа: https://docs.microsoft.com/ru-ru/xamarin.

2. Strategy Analytics Wearable Device Ecosystems (WDE) service report [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.strategyanalytics.com/strategy-analytics/blogs/devices/wearables/wearables/2019/02/28/apple-watch-captures-half-of-18-million-global-smartwatch-shipments-in-q4-2018.

ОБЗОР РЕШЕНИЙ СТАТИСТИЧЕСКОГО АНАЛИЗА КОДА

Жих А.В.

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

Парамонов А.И. – к.т.н.,доцент

В данной статье рассмотрены проблемы, решаемые с помощью статического анализа кода, а также проанализированы характерные особенности статических анализаторов кода на примере компилятора Roslyn. Проблема статического анализа кода является актуальной, так как языки программирования, как и относящиеся к ним технологии, продолжают развиваться, масштабы проектов увеличиваются, вместе с ними и объем самого кода, а, следовательно, и кодревью усложняется. Статический анализ кода является неотъемлемой частью процесса разработки.

Статический анализ кода — анализ программного кода, который производится без его выполнения. Статический анализ необходимо рассматривать в качестве автоматизированного процесса обзора кода [1]. Кодревью является одним из самых полезных методов для выявления дефектов программного кода. В процессе обзора кода выявляются ошибки, которые могут привести к некорректному поведению. Основным недостатком кодревью является то, что данный процесс трудоёмкий и времязатратным, так как полностью производится человеком, а, следовательно, и дорогостоящий. Говоря о статический анализе, следует подчеркнуть, что он позволяет найти большинство ошибок по мере их появления в коде, а, следовательно, и своевременно их исправить. Данная особенность приводит к улучшению качества самого кода. Следует учитывать и то, что чем раньше ошибка была выявлена, тем меньше будет стоить ее исправление. Статический анализ кода может быть использован, как в своём стандартном виде, с целью поиска дефектов кода, так и с целью обучения либо же более усложненный вариант - проверка кода с точки зрения архитектурного решения. Преимущества статического анализа кода в данных направлениях являются очевидными. Возможность поиска дефектов для них полностью сохраняется только будут усложнены правила распознания того, что является дефектом.

На данный момент находится в использовании широкий спектр анализаторов кода на базе статического анализа, основная специфика которых заключается в том, что они заточены под конкретный язык программирования. Рассмотрим основные особенности реализации статических анализаторов для c# на примере компилятора Roslyn.

Roslyn — это не просто компилятор, а целая платформа, которая разрабатывается корпорацией Microsoft. [2] Она предоставляет возможности для разбора кода и его последовательного анализа. Roslyn также является основой для других анализаторов кода. С помощью данного компилятора можно производить детальный разбор кода для дальнейшего анализа всех поддерживаемых языковых синтаксических конструкций. Коданализ проходит в несколько этапов. Вначале идет сбор необходимых данных, что включает в себя получение проектного решения, затем компиляции и разбор полученных файлов, на основе чего строится неизменяемое дерево синтаксиса и семантическая модель. Roslyn строит дерево синтаксиса на основе программного кода, полученного из проектного файла, состоящее из синтаксических узлов, лексем и дополнительной информации. [3] Полученное дерево используется для анализа конструкций языка. По дереву синтаксиса и происходит перемещение во время анализа, которое осуществляется с помощью рекурсии. Синтаксические узлы содержат основные синтаксические языковые конструкции. На синтаксических узлах основываются

диагностические правила. Основной задачей анализатора кода становится обработка синтаксических узлов. У каждого узла есть свой тип, что упрощает навигацию по самому дереву. Лексемы — узлы дерева, неподлежащие дальнейшему разложению. К ним относят такие элементы, как идентификаторы, специальные символы, а также ключевые слова. Дополнительная информация – это такие узлы дерева, которые после компиляции не будут прорастать в IL-код, например, комментарии. Семантическая модель создается в результате компиляции с использованием синтаксического дерева. Семантическая модель содержит в себе информацию о всех объектах. Более того, оно хранит в себе и информацию о типах этих объектов, что даёт возможность проводить более комплексный анализ.

Процесс статического анализа кода всё еще не способен справляться с более сложными задачами, к примеру, прогнозирование результатов проектной архитектуры и предложения путей коррекции.

Список использованных источников:

1."Статический анализ кода"[Электронный ресурс] – Режим доступа: https://www.viva64.com/ru/t/0046/– Дата доступа: 14.03.2019.

ОСОБЕННОСТИ РАЗРАБОТКИ WEB-ПРИЛОЖЕНИЙ ПРИ НИЗКОМ УРОВНЕ КОМПЬЮТЕРНОЙ ГРАМОТНОСТИ ПОЛЬЗОВАТЕЛЕЙ

Жлобич А.В.

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

Куликов С.С. – к.т.н.,доцент

Современный тенденции ведут к тому, что большую часть рынка завоёвывает web-сфера из-за своей простоты в использовании. Рядовому пользователю привычно понятие сайта и он с лёгкостью использует web-технологии в повседневной жизни. Тем не менее, существуют проекты (ситуации), когда web-приложение используется людьми, далёкими даже от понятия «компьютер», в связи с чем существуют особенности разработки сайтов для такой аудитории.

Чаще всего такие ситуации встречаются на стыке двух сфер: когда программные продукты используются для автоматизации процессов в сфере образования, медицины и т.п., где часто встречаются люди, непривыкшие к работе на компьютерной технике. Их знания и умения часто ограничиваются настольными программами семейства Microsoft Office и браузером.

Таким образом, в случае разработки программного обеспечения для данной аудитории существуют проблемы и особенности, которые будут рассмотрены на примере web-сферы:

1) Устаревшие браузеры. Несмотря на то что большинство браузеров обновляется автоматически, в данном сегменте гораздо чаще можно встретить случаи использования устаревших версий браузеров и редко встречающихся браузеров в принципе. Сложность проекта возрастает, потому что нельзя доверять усреднённой статистике по использованию браузеров, приходится отказываться от новых технологий и возможностей только для поддержания совместимости. Дополнительным минусом является тот факт, что вынужденное использование во frontend-части методов, помеченных как *deprecated*, плохо сказывается на продвижении сайтов в поисковых системах;
2) Вопросы безопасности. Компьютеры пользователей с низким уровнем компьютерной грамотности чаще содержат в себе вредоносные программы, что вынуждает принимать дополнительные меры по обеспечению безопасности;
3) Понятность интерфейса.
   - Стоит избегать большого кол-ва изображений. Сайт должен выглядеть максимально просто;
   - Меню сайта должно быть унифицированным и обеспечивать доступ к большей части сайта;
   - Некоторая часть пользователей данного сегмента не имеет представления о разделении на вкладки и такой стандартной возможности браузеров, как «вернуться назад». В связи с этим не рекомендуется открывать ссылки в новом окне;
   - Если предполагается управление некими данными или ресурсами, во-первых, для любой операции должно запрашиваться подтверждение, во-вторых, нельзя доверять решению пользователя: любые даже подтверждённые изменения, включая удаления, должны быть отменяемыми. Приветствуется сохранение резервных копий без участия пользователя.
4) Шрифт. Не приветствуется использование нестандартных шрифтов. Также стоит учитывать вероятность того, что сайтом будут пользоваться люди с плохим зрением. Существует два