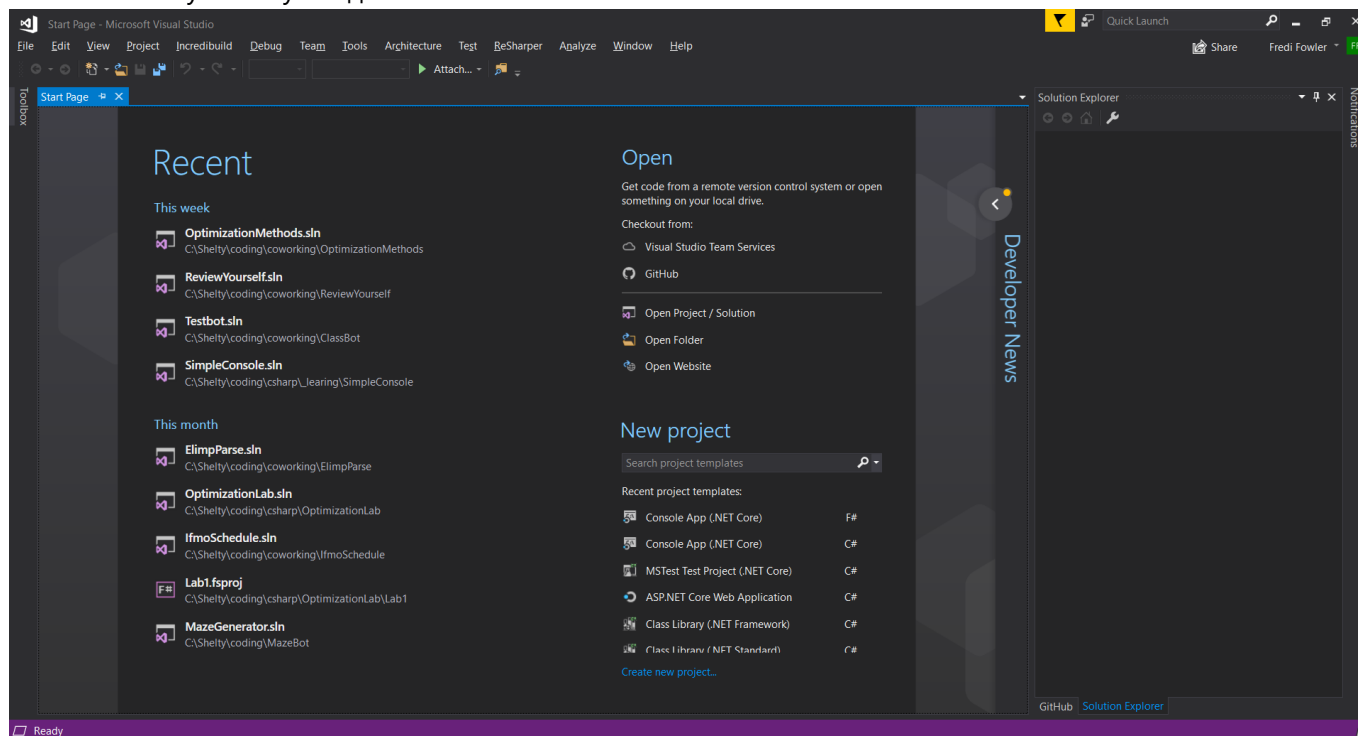


**Цель работы** - изучить используемую IDE, выделить основные элементы среды, которые автоматизируют определенные процессы при разработке.

## Описание IDE

**Visual Studio** - интегрированная среда разработки (IDE) предоставляемая Microsoft. Данная IDE разрабатывается во многом для фреймворка **.NET** и его языков (C++/CLI, C#, F#), но также имеет поддержку и тулзы для работы с SQL, Python, Javascript/Typescript, Java etc.

Важно отметить, что используется именно **VisualStudio** для Windows. **Microsoft** выпустили IDE с таким же названием под **MacOS**, но это не аналогичная IDE, а переделанная XamarinStudio, которая не имеет большинства упомянутых далее компонентов.



## Настройка IDE

Как и любая другая IDE, VisualStudio имеет ряд настроек для повышения гибкости и удобства в использовании.

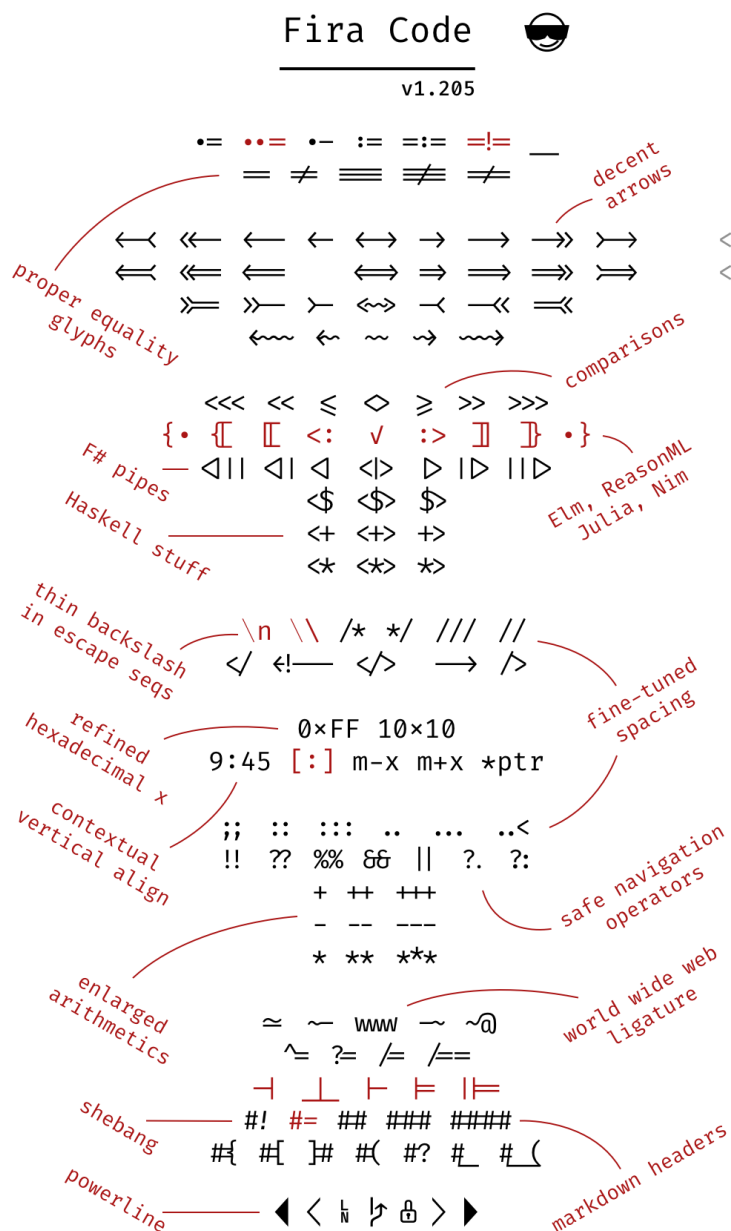
### Учетная запись и синхронизация

VS работает с учетными записями Microsoft. Это упрощает дальнейшую работу со сторонними ресурсами (например, Azure). Помимо этого, настройки IDE синхронизируются с учетной записью и автоматически подтягиваются при авторизации в только что установленной студии. Также, все настройки IDE можно импортировать и экспортировать, что позволяет очень легко настроить рабочее окружение для команды.

### Шрифт

Очень важным для любой среды разработки являются шрифты. VS позволяет без проблем изменять и ставить любой шрифт, доступный в системе. На просторах GitHub можно найти примеры шрифтов

специально разработанных для написания код, например, Fira Code [1].



## Хоткеи

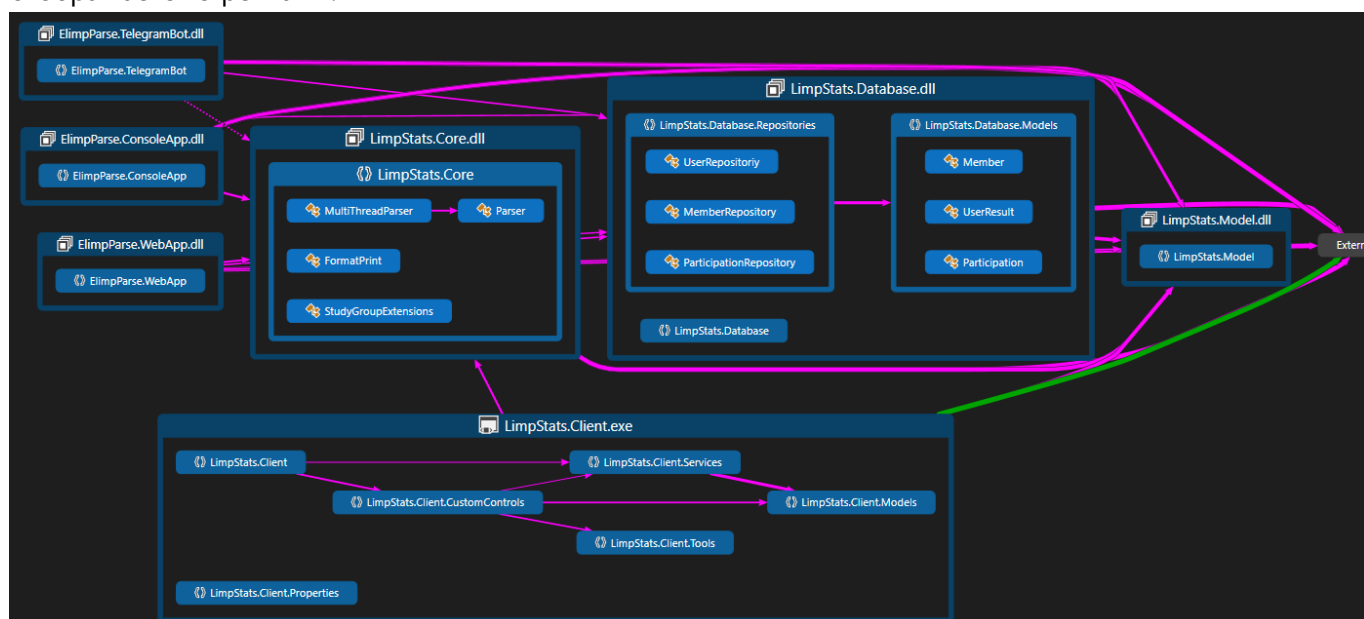
VS предоставляет тулзы для полной кастомизации хоткеев, но почти всегда используется один из двух наборов: стандартные хоткеи VS [2] или хоткеи, которые предоставляет ReSharper [3]. Рассмотрим базовые команды, которые можно выполнить с помощью хоткеев:

- Build.BuildSolution - Ctrl+Shift+B
- Build.Cancel - Ctrl+Break
- Debug.Start - F5
- Debug.StopDebugging - Shift+F5
- Debug.RunToCursor - Ctrl+F10
- Debug.StartWithoutDebugging - Ctrl+F5
- Debug.StepInto - F11
- Debug.StepIntoCurrentProcess - Ctrl+Alt+F11
- Debug.StepOut - Shift+F11

- Debug.StepOutCurrentProcess - Ctrl+Shift+Alt+F11
- Debug.StepOver - F10
- Debug.StepOverCurrentProcess - Ctrl+Alt+F10
- Edit.Copy - Ctrl+C (если не выделен текст, то копируется вся строка, на которой расположен курсор)
- Edit.Cut - Ctrl+X (если не выделен текст, то вырезается вся строка, на которой расположен курсор)
- Edit.Paste - Ctrl+V (если в буфере лежит строка, то она вставляется перед строкой с курсором)
- Edit.CycleClipboardRing - Ctrl+Shift+V (буфер всех скопированных данных)
- Edit.Undo - Ctrl+Z
- Edit.Redo - Ctrl+Y
- Edit.CommentSelection - Ctrl+K, Ctrl+C
- Edit.UncommentSelection - Ctrl+K, Ctrl+U
- File.SaveSelectedItems - Ctrl+S
- File.SaveAll - Ctrl+Shift+S
- Project.AddNewItem - Ctrl+Shift+A
- Refactor.ExtractInterface - Ctrl+R, Ctrl+I
- Refactor.Rename - Ctrl+R, Ctrl+R
- Refactor.ReorderParameters - Ctrl+R, Ctrl+O
- Code cleanup - Ctrl + E, C





















## Визуализация и рефакторинг

Одним из самых важных фишек любой IDE - возможность в удобном виде воспринимать проект, а также его рефакторить. Для этого в VS есть такой инструмент как "**Code map**" - он автоматически генерирует карту зависимостей в проекте. На ней весь код проекта группируется по сборкам, пространствам имен и классам создавая нужную вложенность, а зависимости, вызовы и ссылки между элементами отображаются стрелками.



Одним из инструментов для анализа кода является "**Code metrics**". Он используется для анализа качества написанного кода, позволяет находить места, которые подлежат рефакторингу. С описанием критериев

можно ознакомиться на официальном сайте Майкрософт [4]. Пример:

Code Metrics Results					
Filter: <span>None</span>		Min: <span></span>	Max: <span></span>		
Hierarchy <span>▲</span>		Maintainability Index	Cyclomatic Complexity	Depth of Inheritance	Class Coupling
<span>▲</span>  ReviewYourself (Debug)		84	1 013	3	266
▶  ReviewYourself		79	12	2	59
▶  ReviewYourself.Areas.HelpPage		73	367	2	137
▶  ReviewYourself.Areas.HelpPage.		80	12	3	20
▶  ReviewYourself.Areas.HelpPage.		90	135	3	71
▶  ReviewYourself.Areas.HelpPage.		90	23	1	10
▶  ReviewYourself.Controllers		74	76	3	35
▶  ReviewYourself.Models		93	88	1	7
▶  ReviewYourself.Models.Repositc		100	38	0	8
▶  ReviewYourself.Models.Repositc		66	122	1	20

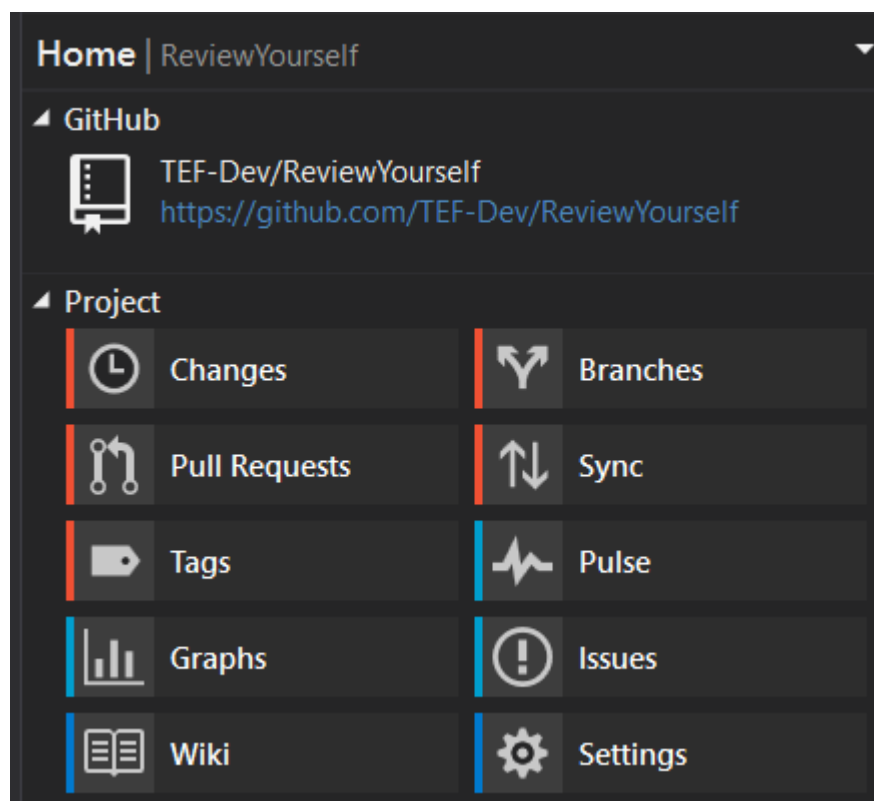
Для более подробного анализа можно воспользоваться "Code issues". Это инструмент, который анализирует код и выводит информацию о всех возможных ошибках и предупреждения в коде:

Inspection Results - Code Issues in 'ReviewYourself'	
Code Issues in 'ReviewYourself'	
Group by: Issue Category	
Type to search	
Displaying 323 of 549 found issues	
Common Practices and Code Improvements (10 items)	
Compiler Warnings (1 item)	
Code is unreachable	
Constraints Violations (1 item)	
Namespace does not correspond to file location, should be: 'ReviewYourself.Models.Tools.DataReccordExtensions'	
Potential Code Quality Issues (43 items)	
Class 'Config' is never instantiated	
Class 'Comment' is never instantiated	
Class 'ReviewCriteria' is never instantiated	
Class 'FilterConfig' is never instantiated	
Class 'BundleConfig' is never instantiated	

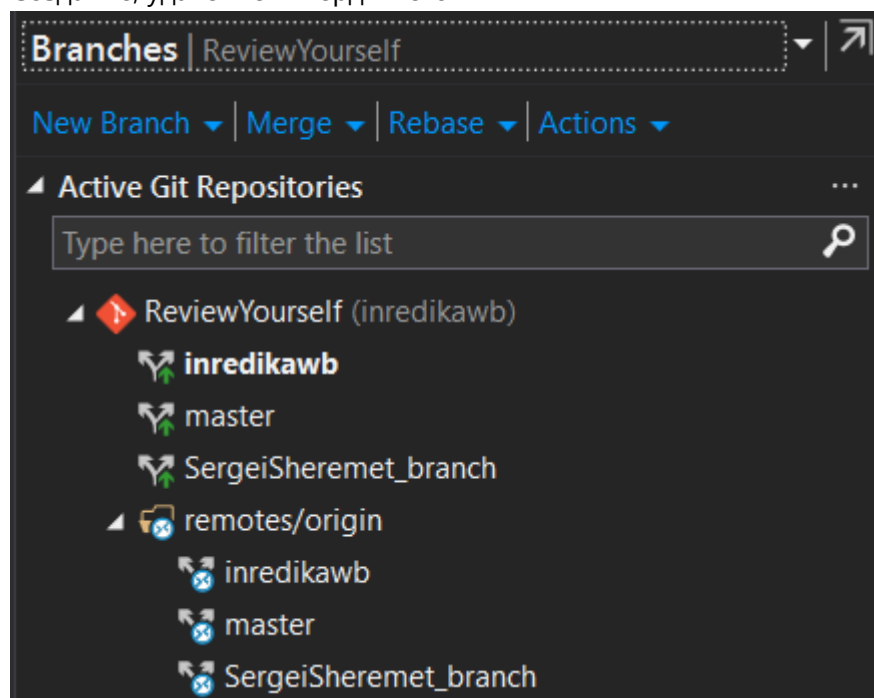
## VCS

В Visual Studio есть поддержка git и VSTS. Работа с git очень упрощается за счет встроенного UI, которым можно выполнить большинство базовых задач git'a не прибегая к консоли:

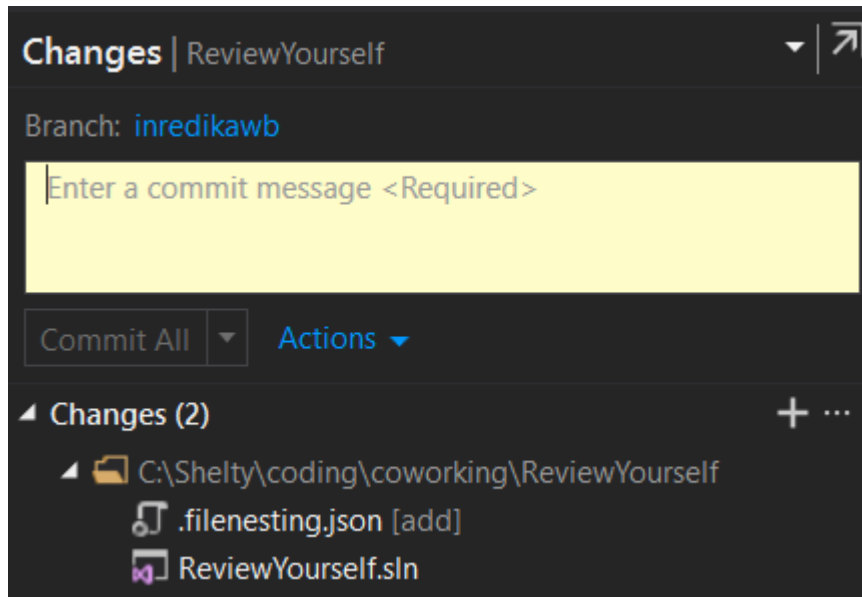
- Просмотр информации о репозитории с возможностью перейти сразу в нужную вкладку на GitHub



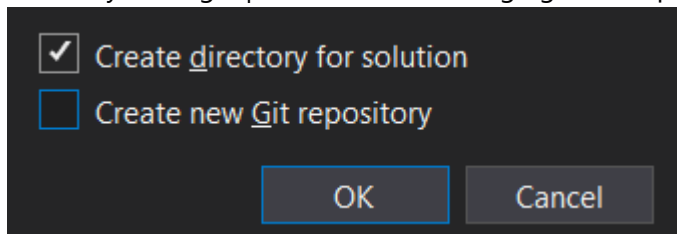
- Создание, удаление и мердж веток



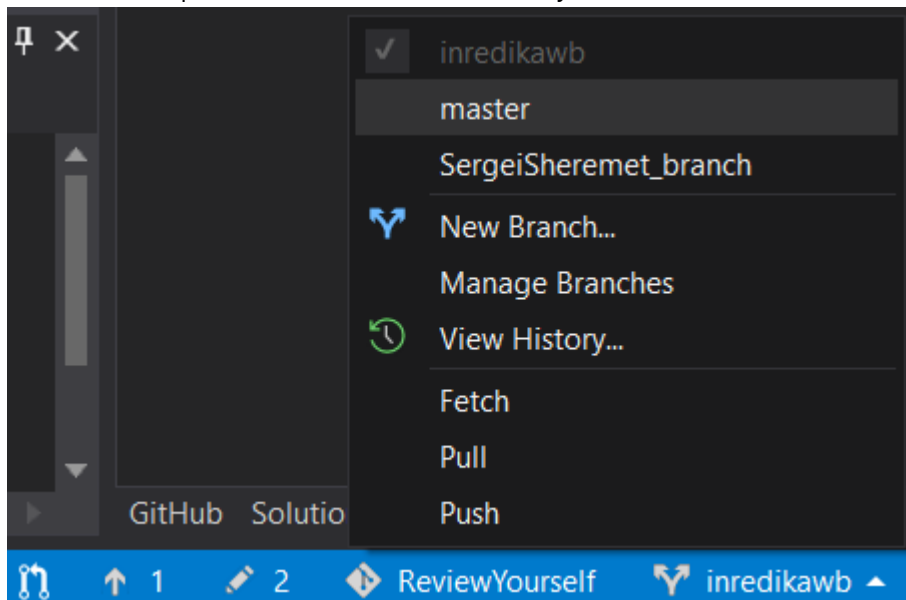
- Комит изменений



При создании нового проекта, можно сразу же инициализировать. Студия автоматически создаст в папке нужные git файлы, в том числе .gitignore с прописанными правилами под Visual Studio:



Во время работы с проектом, который является git-репозиторием, на панели отображается информация о текущем репозитории, рабочей ветке с возможностью ее тут же изменить, количестве измененных файлов, а также новых не запущенных комитов.



Также, по каждому методу Visual Studio пишет все изменения из git-a.

Commit ID	Description	Author	Date	
d7e84f44	Completed main functionality of DB api -Completed Token...	Sergei (sheremet5555@gm...	21.05.2018	
050b5b97	Almost completed ReviewRepo & fixed other repos	Sergei (sheremet5555@gm...	20.05.2018	
36b1f61b	Solution/Review controllers tests and minor fix	Inredika WhiteBlack (inredik...	20.05.2018	
3cdc236c	Review implementation and some minor fix	Inredika WhiteBlack (inredik...	20.05.2018	
a101f35a	Add repositories and services interfaces	Inredika WhiteBlack (inredik...	12.05.2018	
<a href="#">Show all file changes</a>			Changes in months: 12	

ays ago | 2 authors, 5 changes | 0 exceptions

## Azure DevOps (VSTS/TFS)

У Microsoft есть своя разработка для организации командной работы - **Azure DevOps** (который ранее существовал под названием Visual Studio Team Services(который ранее существовал под названием Visual Studio Online(который ранее существовал под названием Team Foundation Server Cloud))) [5] Это набор большого количества различных тулзов для DevOps и Agile, который интегрирован в Visual Studio:

The screenshot displays the Azure DevOps (VSTS) web interface for a project named 'imwt'. The top navigation bar includes links for Refresh, Project homepage, Repositories, Build pipelines, Release pipelines, Agile backlogs, Users & groups, and Delete. The main content area is divided into two sections: 'CI/CD pipeline' and 'Azure resources'.

**CI/CD pipeline:**

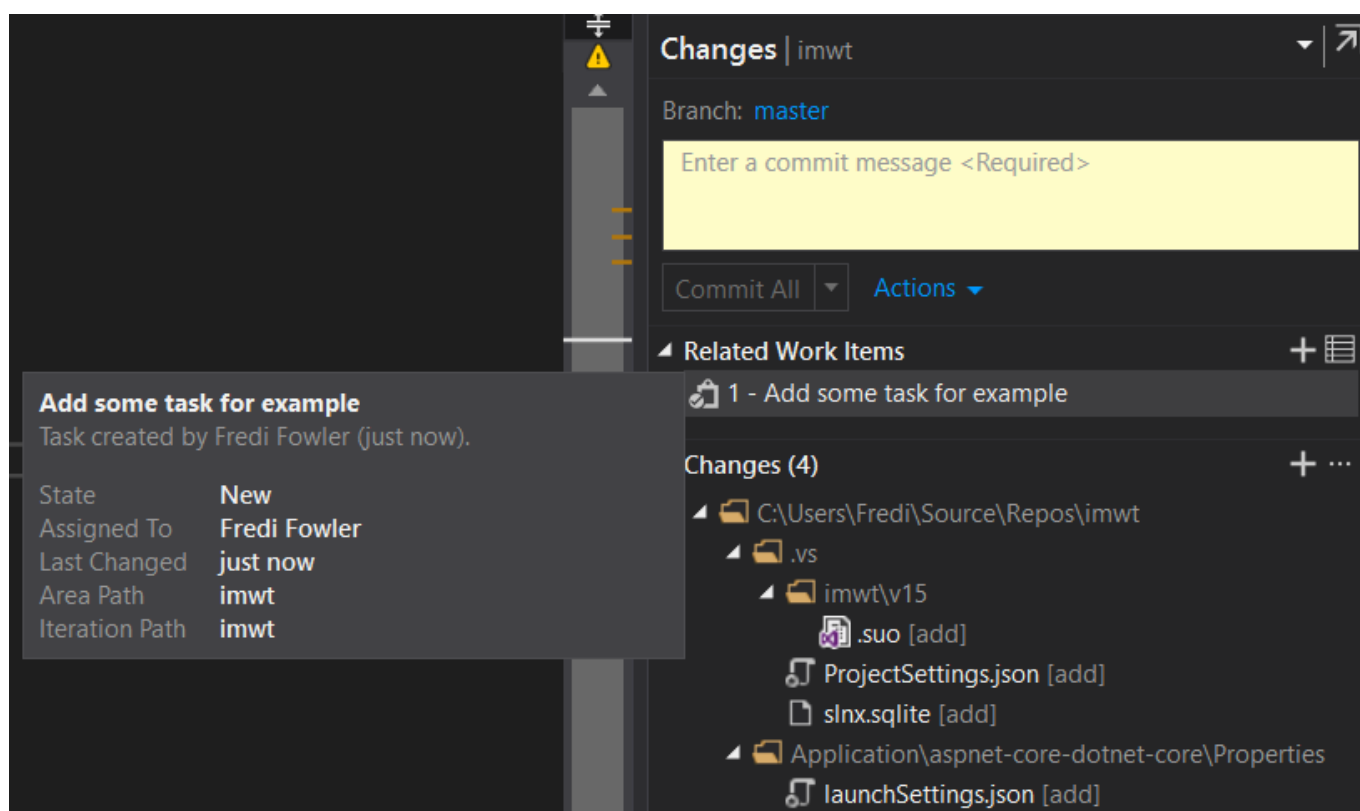
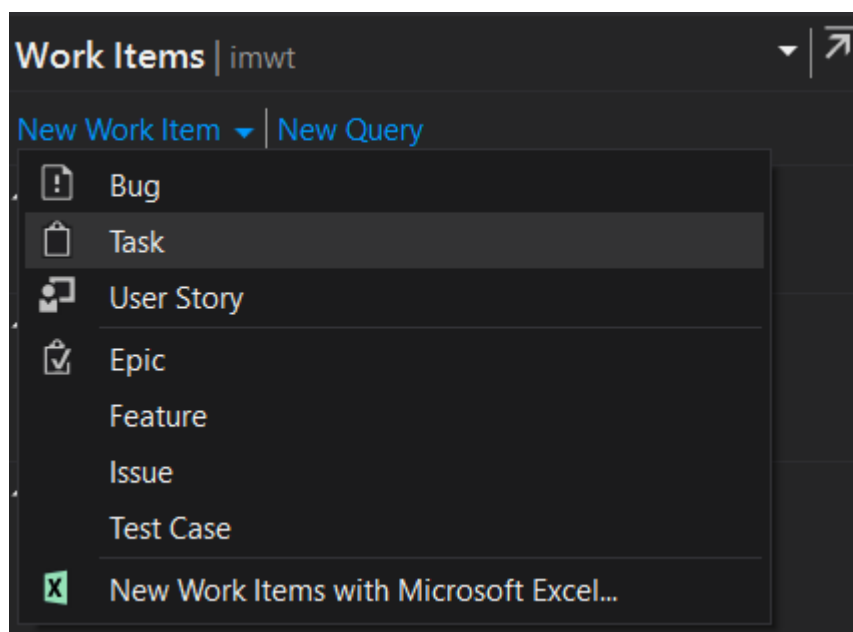
- Code:** Shows the repository 'imwt' with a commit '9a9a18ba' (First commit) by 'azuredevopsproject' 1 day ago.
- Build:** Shows a build 'Build 20180922.1' (Succeeded) 16 minutes ago.
- dev:** Shows a release 'Release-1' (Succeeded) 10 minutes ago.

**Azure resources:**

- Application endpoint:** <http://imwt.azurewebsites.net> (Browse)
- App Service:** 'imwt' (Running)
- SQL Database:** 'imwt-db' (Online)
- Release:** 'Release-1' (Succeeded)

**Application Insights:**

В лучших традициях Agile, в системе (а также в Visual Studio) есть таски, к которым привязываются КОММИТЫ.



## Разработка

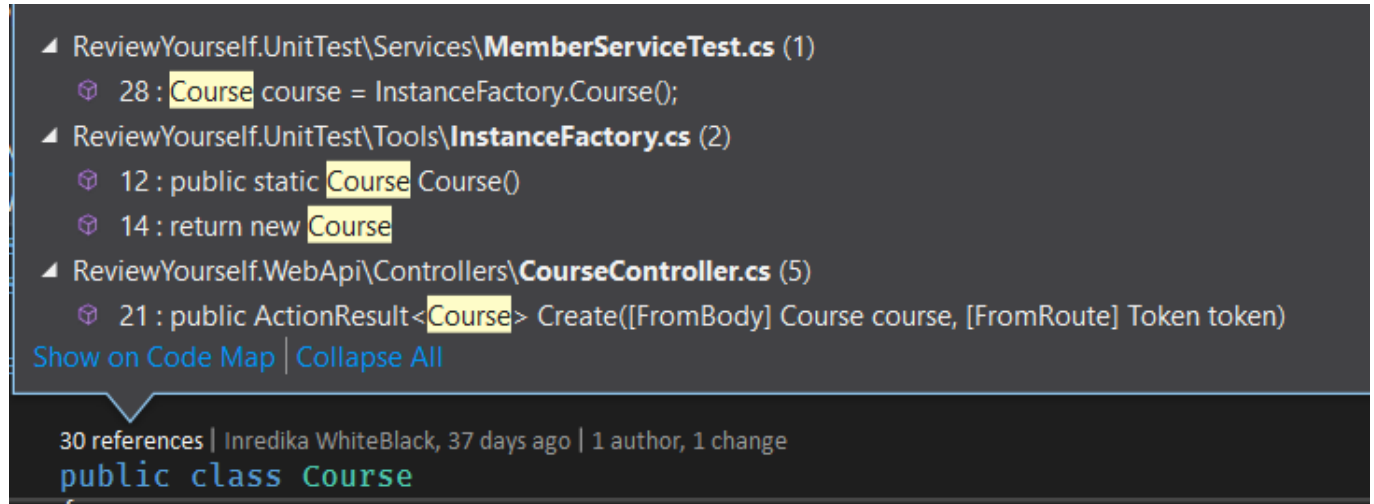
---

### Навигация

Visual Studio предоставляет возможности для удобной навигации по проекту: - Ctrl + клик на тип (класс, структуру, перечисление) - переход к месту определения типа - Для любого типа или его метода можно

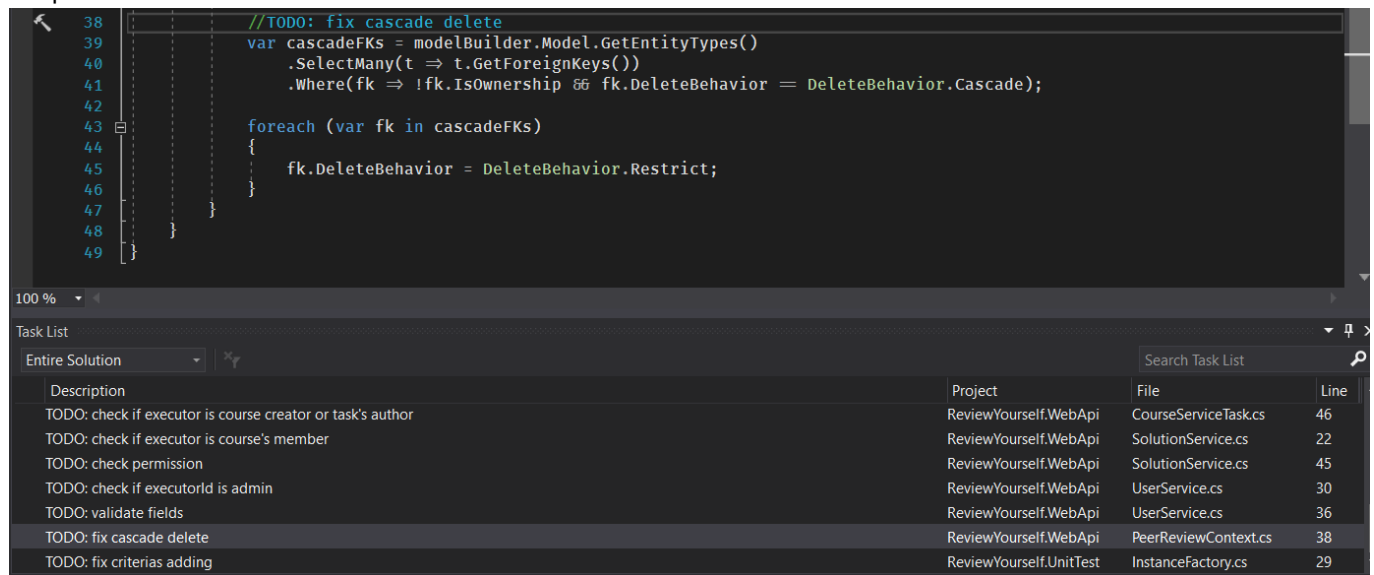


увидеть список референсов (и сразу же открыть карту кода для него):



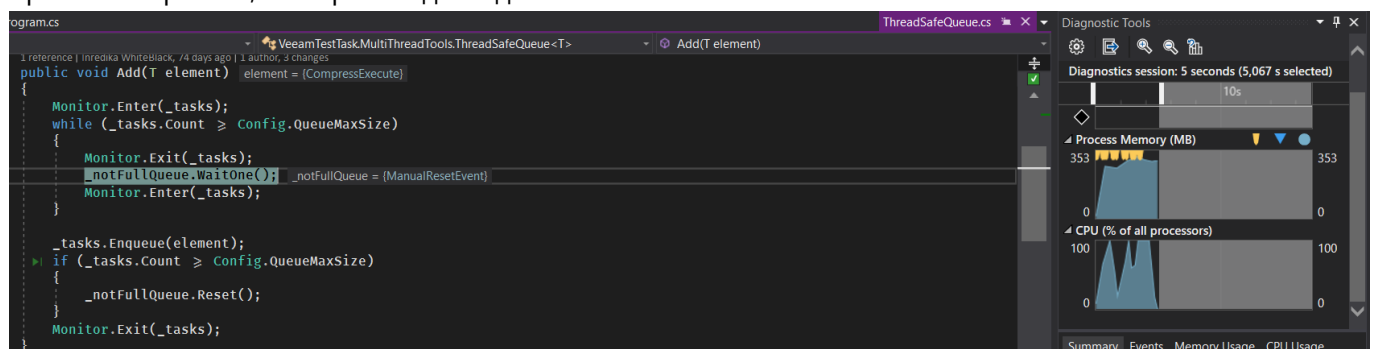
## Task list

Отображение всех комментариев, которые начинаются с "TODO", чтобы их можно было легко найти и открыть.



## Дебаг

Одной из важнейших частей разработки является отладка. Visual studio имеет один из самых функциональных инструментов для этого. Во время выполнения кода можно поставить паузу и проанализировать, что происходит в данный момент:



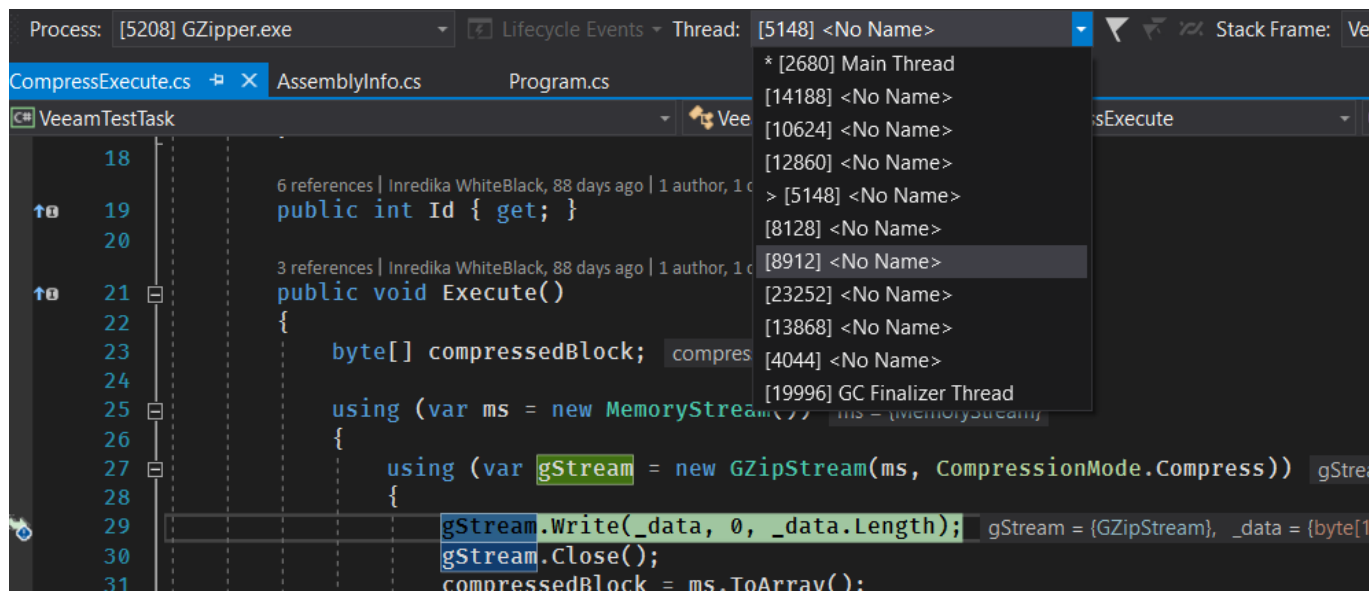
На скрине можно увидеть участок, на котором было остановлено выполнение. Слева отображается дебаг панель, которая выводит информацию о использовании ресурсов (CPU, memory) на протяжении всего периода выполнения. Кроме этого, во время остановки выполнения можно посмотреть

информацию о всех доступных переменных, их значение:

```
private readonly Queue<T> _tasks = new Queue<T>();
private readonly ManualResetEvent _tasksCount = new ManualResetEvent(false);
private readonly ManualResetEvent _tasksCompleted = new ManualResetEvent(false);

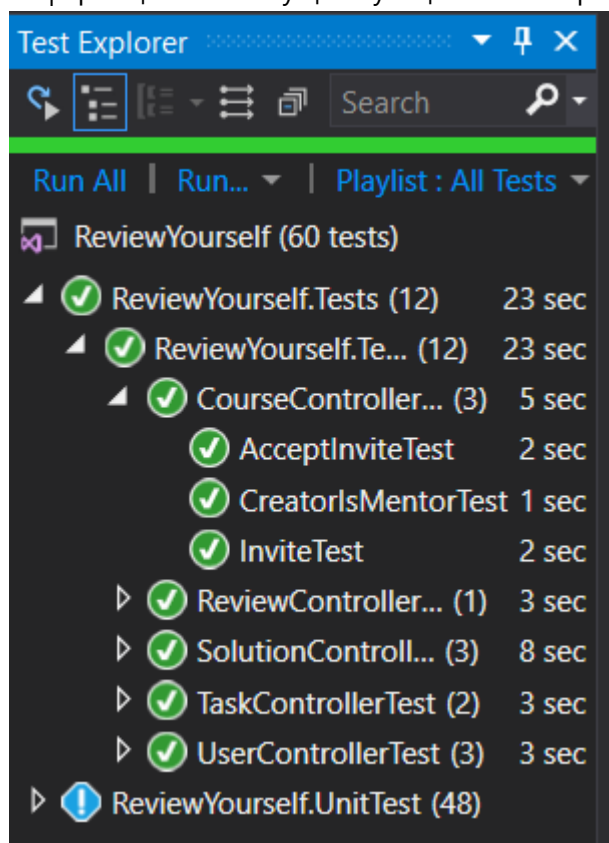
1 reference | Inredika WhiteBlack, 74 days ago | 1 author
public void Add(T element)
{
    Monitor.Enter(_tasks);
    _tasks.Enqueue(element);
    _tasksCount.Increment();
    _tasksCompleted.Reset();
}
```

Если выполняется сразу несколько потоков, то между ними можно переключаться, чтобы увидеть, где они выполняются:

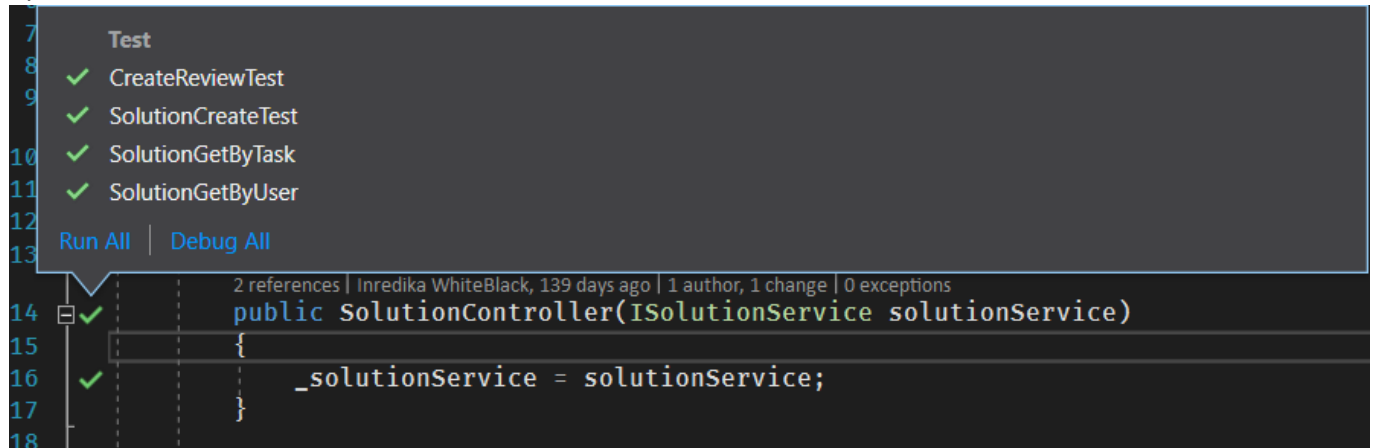


## Unit-test

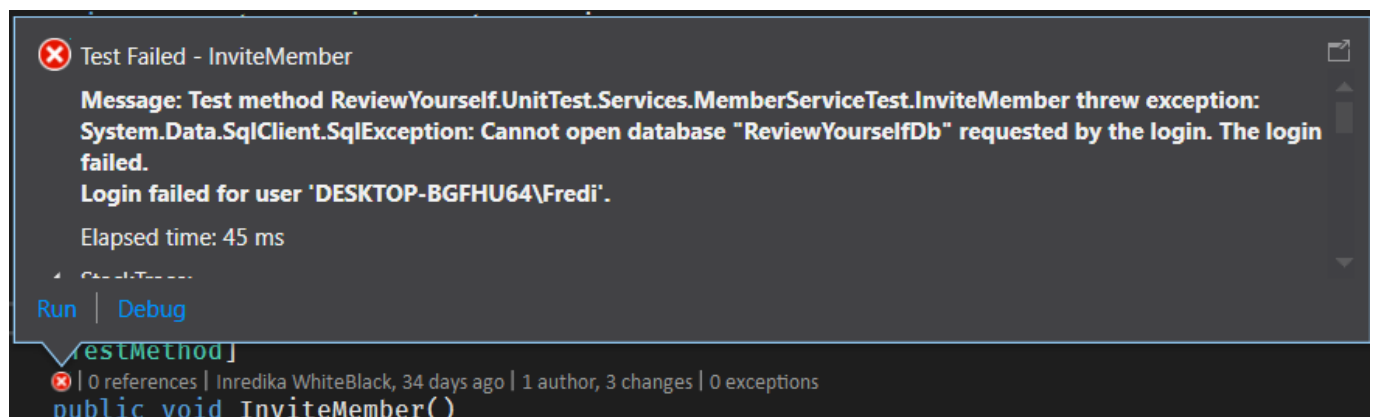
Неотъемлемой частью разработки любого продукта является написание Unit test'ов. Такие тесты нацелены на проверку небольших блоков кода, чтобы проверить не привело ли к возникновению ошибок очередное изменение кода. В Visual Studio есть специальное окно, в котором отображается информация о всех существующих тестах в решении:



Отдельно стоит отметить Live unit testing. Этот инструмент в реальном времени при изменении кода перевыполняет связанные с данным участком юнит-тесты, отображает информацию об успешности их прохождения:



В случае возникновения ошибок в юнит-тестах, над каждым тестом можно узнать информацию о его выполнении:



## Ссылки

1. FiraCode - <https://github.com/tonsky/FiraCode>
2. Default keyboard shortcuts in Visual Studio - <https://docs.microsoft.com/en-us/visualstudio/ide/default-keyboard-shortcuts-in-visual-studio?view=vs-2017>
3. Resharper default keymap - [https://www.jetbrains.com/resharper/docs/ReSharper\\_DefaultKeymap\\_VSscheme.pdf](https://www.jetbrains.com/resharper/docs/ReSharper_DefaultKeymap_VSscheme.pdf)
4. Code metrics - <https://docs.microsoft.com/en-us/visualstudio/code-quality/code-metrics-values?view=vs-2017>
5. Azure DevOps - <https://azure.microsoft.com/en-us/blog/introducing-azure-devops/>