<https://medium.com/@p1t1ch/visual-studio-code-%D0%BD%D0%B0%D1%81%D1%82%D1%80%D0%BE%D0%B9%D0%BA%D0%B0-%D0%B8-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5-%D1%87%D0%B0%D1%81%D1%82%D1%8C-1-7f1a26806522>

**[GitLens — Git supercharged](https://marketplace.visualstudio.com/items?itemName=eamodio.gitlens" \t "_blank)**

GitLens — это многофункциональное расширение, предоставляющее возможности по отслеживанию истории изменений. Оно настолько крупное, что идёт со своим интерфейсом настроек, который можно вызвать через палитру команд. Так что если будете настраивать, то пользуйтесь этим интерфейсом — оттуда убран ряд заумных внутренних команд, которые при необходимости можно найти в основных настройках редактора. Расширение разбито на отдельные группы функционала — пройдёмся по ним:

1. **GitLens Explorer**— проводник по коммитам репозитория. Предоставляет возможность просмотра коммитов, как в текущей ветке, так и в остальных. Через ПКМ можно открыть репозиторий в окне браузера. Данная панель доступна с вкладки GitLens на панели управления, но через настройки её можно перенести в основное меню Source Control.
2. **GitLens File History Explorer** — история коммитов открытого файла. Также можно перенести в Source Control.
3. **Code Lens** — встраиваемая в код через линзы информация по автору и дате последнего изменения для всего документа и/или по отдельным контейнерам. Нахожу её бесполезной и загромождающей соурскод, так что отключил.
4. **Current Line Blame** — при фокусе на отдельной строке показывает линзу с информацией по дате, автору и имени коммита, в котором она была в последний раз изменена. Также отключил, т.к. это отвлекает и в целом бесполезно, т.к. при необходимости можно вывести Blame по всему файлу (об этом дальше).
5. **Gutter Blame**— отображает слева на полях файла информацию для каждой строки по автору, дате и коммиту, в котором эта строка в последний раз менялась. При выборе строки подсвечиваются также все изменения из того же коммита. Также поддерживается Heatmap — цветные индикаторы, которые тем ярче, чем новее строка. Это крайне наглядный интерфейс, который незаменим, если вас интересуют только последние изменения. Если же нужно найти информацию, когда в принципе был добавлен этот участок кода, то он уже не подойдёт — нужно копаться в File History. Эта функция вызывается по запросу (on-demand) через кнопку сверху справа, и в этом её преимущество над всевозможными дотошными линзами.
6. **Hovers** — отображает всплывающее окно с информацией по коммиту с различными кнопками управления. Может активироваться как наведением на строку в режиме Gutter Blame, так и при наведении на линзу в режиме Current Line Blame. Также считаю избыточным.
7. **Status Bar Blame**— добавляет информацию по последнему изменению строки в статус бар. При нажатии по умолчанию вызывает командную строку со списком команд.
8. **Modes** — GitLens имеет возможность настройки режимов, доступных для быстрого переключения через нижнюю панель. По умолчанию доступны режимы Zen и Review — один для разработки без визуальных отвлекающих факторов, а второй для полноценного обзора. Можно в одном из них отключить линзы, блейм для текущей строки, проводники, ховеры и статус бар, а для второго всё включить. В принципе, рабочий концепт, если вам нравится функционал линз, но они не нужны вам при разработки. Но для себя я понял, что мне хватает включения режима аннотаций (Gutter Blame) при необходимости и прогулок по проводникам с историей в более сложных случаях, так что эта тема с режимами будет избыточна.
9. **Command Line**— GitLens предоставляет широкий список команд для фанатов командной строки, но лично я ими не пользуюсь.
10. **Comparison Tools** — GitLens позволяет гулять по разным версиям файла в режиме сравнения двух версий файлов. Режим активируется из интерфейса через кнопку **“Compare File with Previous Revision”** в верхнем правом углу. После активации можно углубляться вглубь истории, пока у вас не закончатся коммиты 😉

Я описал основные функции расширения, а что уже из этого использовать, а что нет — дело ваше. Мою сборку для работы с историей коммитов можно видеть на скриншоте наверху раздела — там активирован режим **Gutter Blame** и открыта вкладка Source Control, на которой можно видеть **GitLens Explorer** и **Gitlens File History Explorer**. Также в статус баре можно видеть индикатор **Status Bar Blame** и сверху справа рядом с горящей кнопкой, активирующей Gutter Blame, кнопку, запускающую Comparison Tools. Остальные перечисленные функции отключены.

В итоге все мои настройки этого расширения сводятся к приведению редактора к минималистичному виду — отключения всех визуальных фич и перемещение всех проводников в меню Source Control. Для себя я нахожу все эти линзы и ховеры бесполезными и вместо этого включаю аннотации всего файла при необходимости. Если вам нравится этот функционал, то предложу настроить modes, чтобы они выводились в review-режиме, но не мешались при разработке. Перемещение же всех проводников в дефолтную вкладку позволяет держать всю работу с Гитом на одной панели, что полностью убирает дополнительную вкладку из интерфейса и на практике повышает частоту пользования все данным функционалом и упрощает работу в целом.

“gitlens.codeLens.enabled”: false,  
“gitlens.currentLine.enabled”: false,  
“gitlens.hovers.enabled”: false,  
“gitlens.mode.statusBar.enabled”: false,  
“gitlens.gitExplorer.location”: “scm”,  
“gitlens.historyExplorer.location”: “scm”,  
“gitlens.resultsExplorer.location”: “scm”,