**Что такое тип тестирования?**

В настоящее время нет общепринятого определения «типа тестирования программного обеспечения». Это не редкость, когда методы, уровни или даже тестовая дизайн техника определяются как тип тестирования. Например, иногда white box тестирование, интеграционное тестирование или даже тестирование граничный значений рассматриваются как типы тестирования.

Необходимо обратиться к официальным источникам, чтобы избежать неоднозначных толкований. Международная квалификационная коллегия по тестированию программного обеспечения (International Software Testing Qualifications Board - ISTQB) — это уважаемая международная организация, которая предлагает стандарты в тестировании программного обеспечения, которые признаны во всем мире.

Согласно определению ISTQB, типы тестирования являются **«средством четкого определения цели определенного уровня для программы или проекта»**. Тестер фокусируется на конкретной цели тестирования во время выполнения тестового случая. В зависимости от его целей существует четыре типа тестирования программного обеспечения:

* Функциональное тестирование
* Нефункциональное тестирование
* Структурное тестирование
* Тестирование изменений

Конечно, существует множество подтипов тестирования. Вы можете увидеть их на рисунке ниже:

<https://geteasyqa.com/wp-content/uploads/2017/04/TYPES-OF-SOFTWEAR-TESTING.svg>

## ****Функциональное тестирование****

Функциональное тестирование подтверждает, что каждая функция программного приложения работает в соответствии с требованиями спецификации. Функциональное тестирование показывает **«Что делает система»**. Цель этого тестирования - проверить, является ли система функционально совершенной.

В функциональном тестировании необходимо учитывать пять шагов:

* Подготовка тестовых данных на основе функций в спецификации
* Бизнес-требования являются вкладом в функциональное тестирование
* Вывод результатов работы на основе функциональных спецификаций;
* Выполнение тестовых случаев
* Соблюдайте фактические и ожидаемые результаты.

Существуют две перспективы, в которых функциональность тестирования может быть выполнена на основе тестирования на основе требований и тестирование на основе бизнес-процессов.

**Тестирование на основе требований** выполняется в строгом соответствии с установленными требованиями.

**Тестирование на основе бизнес-процесса** выполняется в соответствии с знаниями, основанными на повседневном использовании системы в бизнесе.

**Преимущества функционального тестирования:**

* Функциональное тестирование имитирует фактическое использование системы.
* Оно выполняется в условиях, близких к клиентским.
* Легко провести ручное тестирование.

**Ограничения функционального тестирования:**

* Существует высокая вероятность избыточного тестирования.
* Логические ошибки в программном обеспечении могут быть пропущены при обеспечении функционального тестирования.

Легко найти и использовать инструменты для функционального тестирования. Наиболее известными из них являются: **[Selenium](http://www.seleniumhq.org/%22%20%5Ct%20%22_blank)** (как веб-приложения, так и настольные приложения), **[Robotium](https://robotium.com/%22%20%5Ct%20%22_blank)** (приложение для Android), **[Linux Test Project](https://linux-test-project.github.io/%22%20%5Ct%20%22_blank)**, **[JUnit](http://junit.org/junit4/%22%20%5Ct%20%22_blank), [Sprinter by Hewlett Packard Entreprise](https://saas.hpe.com/ru-ru/software/sprinter-manual-software-testing%22%20%5Ct%20%22_blank)** (ручное тестирование), **[Browserstack](http://www.browserstack.com/%22%20%5Ct%20%22_blank)** (как автоматическое, так и ручное тестирование), **[Usersnap](https://usersnap.com/%22%20%5Ct%20%22_blank)** (ручное тестирование).

## ****Нефункциональные типы тестирования****

**Нефункциональные типы тестирования** связаны с нефункциональными требованиями. Нефункциональное тестирование помогает оценить готовность системы в соответствии с различными критериями, которые не охватываются функциональным тестированием. В отличие от функционального тестирования, он показывает «**Как хорошо работает система**».

Давайте рассмотрим вариации многотипных подтипов нефункционального тестирования.

**Тестирование пользовательского интерфейса**(User Interface - UI) предназначено для обеспечения соответствия графического пользовательского интерфейса приложения требованиям. Это помогает оценить элементы дизайна, такие как макет, цвета, шрифты, размеры шрифта, метки, текстовые поля, форматирование текста, титры, кнопки, списки, значки, ссылки и контент.

Подходы к тестированию пользовательского интерфейса: руководство, запись и ответ, основанные на модели.

Самые популярные специальные инструменты и рамки для тестирования пользовательского интерфейса: **[FitNesse](http://www.fitnesse.org/), [iMacros,](https://imacros.net/download/) [Coded UI,](https://msdn.microsoft.com/en-us/library/dd286726.aspx) [Jubula,](https://www.eclipse.org/jubula/) [LoadUI](https://www.loadui.org/).**

**Тестирование опыта взаимодействия** (User Experience - UX) нацелено на проверку взаимодействия с продуктами и услугами компании. Собственно, UX намного больше, чем визуальный интерфейс вашего продукта. Он содержит:

* Впечатления, которые клиент отвлекает от взаимодействия с продуктом или услугой.
* Процесс, который должен пройти клиент, чтобы узнать продукт или услугу компании.
* Последовательность действий клиента, когда он взаимодействует с интерфейсом.

Специальные инструменты и рамки для тестирования UX: **[Usabilla](https://www.usabilla.com/%22%20%5Ct%20%22_blank), [Omnigraffle](https://www.omnigroup.com/omnigraffle%22%20%5Ct%20%22_blank), [Visual Web Optimizer](https://vwo.com/%22%20%5Ct%20%22_blank), [UXPin](https://www.uxpin.com/%22%20%5Ct%20%22_blank), [Crazy Egg](https://www.crazyegg.com/%22%20%5Ct%20%22_blank).**

**Тестирование хранилища** проверяет тестируемое приложение, сохраняет соответствующие данные в правильных каталогах и имеет достаточное пространство для предотвращения неожиданного завершения из-за недостаточного дискового пространства. Это помогает определить, использует ли приложение больше памяти, чем предполагалось, поскольку заполнение дискового пространства может привести к значительным простоям.

Инструменты часто используются для тестирования хранилища: **[HCIbench](https://labs.vmware.com/flings/hcibench%22%20%5Ct%20%22_blank), [Iometer](http://www.iometer.org/%22%20%5Ct%20%22_blank), [Diskspd Utility](https://gallery.technet.microsoft.com/DiskSpd-a-robust-storage-6cd2f223%22%20%5Ct%20%22_blank).**

**Операционное тестирование** направлено на оценку системы или компонента в его операционной среде. Используя его, мы можем обеспечить соответствие системы и компонентов в стандартной операционной среде приложения.

Это тестирование в основном фокусируется на оперативной готовности системы, которая, как предполагается, имитирует производственную среду.

**Тестирование безопасности** направлено на то, чтобы обеспечить информационную систему защитой данных и поддерживать функциональность по назначению. Тестирование на проникновение и тестирование уязвимостей являются видами типов тестирования безопасности.

**Тестирование проникновения** — это симуляция атаки вредоносного источника, которое позволяет оценить безопасность компьютерной системы или сети.

**Тестирование уязвимостей** направлено на оценку кванта рисков, связанных с системой, чтобы уменьшить вероятность. Это помогает предотвратить проблемы, которые могут повлиять на целостность и стабильность приложения.

Такие инструменты, как **[Retina CS Community](https://www.beyondtrust.com/products/retina-cs-community/%22%20%5Ct%20%22_blank),**[**OWASP Zed Attack Proxy**](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)**, [Veracode,](https://www.veracode.com/%22%20%5Ct%20%22_blank) [Google Nogotofail,](https://github.com/google/nogotofail%22%20%5Ct%20%22_blank)**и[**SQL Map**](http://sqlmap.org/)часто используются для тестирования безопасности.

**Тестирование конфигураци**и выполняется для проверки системы с каждой из поддерживаемых программных и аппаратных конфигураций:

* Конфигурация OС - Win 7 32 bit/64 bit, Win 8 32 bit/64 bit, Win 7 32 bit/64 bit, Win 8 32 bit/64 bit.
* Конфигурация базы данных - Oracle, DB2, MySql, MSSQL Server, Sybase.
* Конфигурация браузера - IE 10, IE 11, Mozilla Firefox, Google Chrome.

Существуют различные виды тестирования конфигурации, такие как **тестирование конверсий**, **кросс-браузерное тестирование**, **тестирование двоичной портативности** и **кроссплатформенное тестирование**.

**Тестирование конверсий** обеспечивает правильное преобразование данных из существующих систем для использования в системах замещения.

**Кросс-браузерное тестирование** выполняется для проверки правильной работы приложения или системы в разных конфигурациях браузера: Mozilla Firefox, Google Chrome, Internet Explorer и Opera и т. Д.

В отличие от кросс-браузерного тестирования, **кроссплатформенное тестирование** нацелено на оценку работы приложения в разных ОС: Windows, iOS / Mac OS, Linux, Android и BlackBerry и т. Д.

**Тестирование двоичной портативности** позволяет оценить переносимость программного обеспечения, выполнив программное обеспечение на разных платформах и в среде. Он используется для подтверждения спецификации прикладного двоичного интерфейса (Application Binary Interface - ABI).

Основные инструменты, которые часто используются для тестирования всех конфигураций:**[BrowserStack,](https://www.browserstack.com/%22%20%5Ct%20%22_blank)**[**CrossBrowserTesting by Smart Bear**](https://crossbrowsertesting.com/)**, [Litmus](https://litmus.com/%22%20%5Ct%20%22_blank)**[**, Browsera**](https://www.browsera.com/)**, [Rational Clearcase by IBM](https://www-03.ibm.com/software/products/ru/clearcase%22%20%5Ct%20%22_blank), [Ghostlab](https://vanamco.com/ghostlab/%22%20%5Ct%20%22_blank).**

**Тестирование локализации** выполняется для адаптации глобализированного приложения к определенной культуре / языку. Этот процесс включает перевод всех строк родного языка на целевой язык и настройку графического интерфейса, чтобы он соответствовал целевому рынку. **Тестирование на глобализацию** и **тестирование на интернационализацию** являются одними из его видов.

Проверка на предмет глобализации проверяет правильную функциональность продукта с любыми настройками культуры / локализации с использованием любого типа международного ввода.

Тестирование на интернационализацию проверяет правильность содержимого контента на разных языках и местах.

Локализация обычно осуществляется с использованием некоторой комбинации внутренних ресурсов, независимых подрядчиков и полномасштабных услуг компании локализации. Вот некоторые инструменты для обеспечения тестирования локализации – **[eggPlant](https://www.testplant.com/eggplant/testing-tools/%22%20%5Ct%20%22_blank)**, [**Babylon.NET by Redpin**](http://www.redpin.eu/)**,**и **[smartCAT.](https://www.smartcat.ai/lsp/?culture=en" \t "_blank)**

**Тестирование производительности** (Performance testing) намеревается определить, как система работает с точки зрения быстроты реагирования и стабильности при определенной нагрузке. Виды тестирования производительности: **стресс-тестирование** (Stress testing), **нагрузочное тестирование** (Load testing), **тестирование стабильности** (Stability testing), **объемное тестирование** (Volume testing), **тестирование паралелизма** (Concurrency testing), **тестирование на масштабируемость** (Scalability testing), **тестирование на выносливость** (Endurance testing) и т. Д.

**Стресс-тестирование** оценивает поведение системы в пределах ее ожидаемой рабочей нагрузки или выходит за ее пределы.

Проведено **тестирование нагрузки** для оценки поведения системы при увеличении рабочей нагрузки.

**Тестирование на стабильность** направлено на то, чтобы проверить, может ли приложение непрерывно работать в пределах или чуть выше допустимого периода.

**Тестирование объема** позволяет анализировать производительность системы за счет увеличения объема данных в базе данных. Оно проверяет, что любые значения могут стать большими с течением времени (например, накопленные счета, журналы и файлы данных), могут быть адаптированы программой и не заставят программу прекратить работу или ухудшить ее работу.

**Тестирование паралелизма** выполняется для выявления дефектов в приложении, когда несколько пользователей входят в приложение. Используя его, мы можем идентифицировать и измерить проблемы в системных параметрах, такие как время отклика, пропускная способность, блокировки / взаимоблокировки или любые другие проблемы, связанные с параллелизмом.

**Масштабируемость** оценивает способность системы расти, увеличивая различные показатели, такие как рабочая нагрузка на каждого пользователя или количество одновременных пользователей или размер базы данных.

**Испытание на выносливость** определяет проблемы, которые могут возникать при длительном выполнении. Он оценивает поведение системы, когда значительная рабочая нагрузка предоставляется непрерывно.

**Тестирование Ramp** — это испытание на выносливость, которое заключается в непрерывном повышении входного сигнала до тех пор, пока система не сломается.

[**Apache JMeter**](https://jmeter.apache.org/)**,**[**HP LoadRunner**](https://saas.hpe.com/en-us/software/loadrunner)**, [Silk Performer from Micro Focus](https://www.microfocus.com/products/silk-portfolio/silk-performer/%22%20%5Ct%20%22_blank), [WebLOAD](https://www.radview.com/%22%20%5Ct%20%22_blank),**и**[Gatling](http://gatling.io/%22%20%5Ct%20%22_blank)**часто используются для выполнения различных видов тестирования производительности.

**Тестирование восстановления** предназначено для оценки способности системы восстанавливаться после сбоев, сбоев оборудования или других катастрофических проблем. Он выполняется группами тестирования.

[**TestDisk**](https://www.cgsecurity.org/wiki/TestDisk_Download)[**, Recuva by Piriform**](https://www.piriform.com/recuva/download)**,**[**Wise Data Recovery by WiseCleaner**](https://www.wisecleaner.com/wise-data-recovery.html) и[**Restoration by Softonic**](https://restoration.en.softonic.com/)

являются основными специальными инструментами для тестирования восстановления.

**Проверка совместимости**проверяет совместимость приложения в разных средах: аппаратное обеспечение, программное обеспечение, операционная система, сетевая среда. Существует два типа такого типа тестирования: **обратная совместимость, тестирование прямой совместимости**.

**Тестирование обратной совместимости** гарантирует, что новая версия продукта продолжит работу с более старым продуктом.

**Тестирование прямой совместимости** обеспечивает подключение к будущей версии продукта.

Для тестирования совместимости могут использоваться бесплатно

[**Browsershots**](https://browsershots.org/)и**MultiBrowser**

**Юзабилити тесты**, выполненные для оценки продукта или услуги путем тестирования его с репрезентативными пользователями. Это помогает определить способность пользователя учиться работать, готовить входные данные и интерпретировать выходы системы или компонента.

**Тестирование на доступность** — это тип тестирования юзабилити, который определяет удобный для пользователя уровень продукта, для людей с ограниченными возможностями (глухих, слепых, умственно отключенных).

Популярные инструменты для тестирования юзабилити: **[User Zoom](https://www.userzoom.com/%22%20%5Ct%20%22_blank), [Reflector,](https://reflector.en.softonic.com/%22%20%5Ct%20%22_blank)** [**Loop11**](https://www.loop11.com/)**.**

## ****Структурное тестирование****

**Структурное тестирование**проверяет реализацию программы или кода посредством тестирования структуры программной системы или ее компонентов. Тестер концентрируется на работе программного обеспечения во время структурных испытаний. Он может использоваться на всех уровнях тестирования.

Основные цели структурного тестирования:

* Очевидная неадекватность идентификации
* Функциональное тестирование
* Чтобы понять, не хватает ли чего в нашем наборе тестов

Методы структурного тестирования:

* **Statement Coverage** проверяет, что каждый оператор в программе выполняется хотя бы один раз во время тестирования программы.
* **Path Coverage** предназначен для удовлетворения критериев охвата каждого логического пути через программу.
* **Branch Coverage** проверяет, имеют ли каждое условие ветвления для программы истинные или ложные значения.
* **Condition Coverage** похоже на Branch Coverage. Основное различие заключается в проверке состояния покрытия для условных и не условных ветвей.

Преимущества структурного тестирования:

* Удаление мертвого кода
* Существует возможность обнаруживать ошибки на ранней стадии.
* Это обеспечивает более тщательное тестирование программного обеспечения.
* Структурное тестирование не является трудоемким процессом.

Недостатки структурного тестирования:

* Структурное тестирование затратное.
* Оно требует знания кода.
* Оно требует глубокого знания инструмента, используемого для тестирования.

Специальные инструменты для структурного тестирования: [JBehave](http://jbehave.org/%22%20%5Ct%20%22_blank), [Cucumber](https://cucumber.io/%22%20%5Ct%20%22_blank), [JUnit](http://junit.org/junit4/%22%20%5Ct%20%22_blank), [Cfix](http://www.cfix-testing.org/%22%20%5Ct%20%22_blank).

## ****Тестирование изменений****

**Тестирование изменений** предоставляется для обеспечения исправления ранее исправленных ошибок и устранения ошибок, которые могут быть случайно отображены в новой версии. В соответствии с этими целями существуют два подтипа тестирования, связанного с изменением: **тестирование подтверждения** (повторное тестирование) и **регрессионное тестирование**.

Очень часто путают тестирование подтверждения с регрессионным тестированием. Как правило, их нужно использовать один за другим. Давайте выясним разницу между этими подтипами тестирования.

Во-первых, вы должны выполнить **тестирование подтверждения**, чтобы убедиться, что ошибка действительно успешно удалена. Проще говоря, тестовый сценарий, который первоначально обнаружил ошибку, выполняется снова, и на этот раз он должен пройти без проблем.

**Регрессионное тестирование** состоит не только из выявленных случаев проверки ошибок. Оно выполняется не только для проверки исправления ошибок, но также для обеспечения того, чтобы новые дефекты не появились или не были обнаружены после изменений.

Надеюсь, приведенная ниже картина поможет вам лучше понять процесс тестирования, связанный с изменениями.

Инструменты для тестирования, связанные с изменением: **[Selenium](http://www.seleniumhq.org/%22%20%5Ct%20%22_blank),**[**HP Quick Test Professional,**](https://saas.hpe.com/en-us/software/uft)[**TestComplete**](https://smartbear.com/product/testcomplete/overview/)**, [TestDrive](https://www.origsoft.com/products/testdrive/%22%20%5Ct%20%22_blank), [SoapUI](http://www.soapui.org/%22%20%5Ct%20%22_blank).**

Теперь у вас есть представление о множестве типов тестирования. Некоторые из них ежедневно используются огромным количеством специалистов, а другие используются очень редко. Во всяком случае, всей жизни недостаточно, чтобы научиться принципиально каждому из типов тестирования. Но практика - лучший способ сделать это. Надеюсь, что представленная здесь информация поможет вам в вашей будущей деятельности в QA. Было бы замечательно узнать ваше мнение о типах тестирования в комментариях.

основные этапы, которые вы должны пройти, чтобы протестировать свой сайт. Они представлены на картинке внизу.



## Тестирование документации

На этом этапе анализируются основные артефакты, связанные с тестированием веб-сайта:

* Требования
* План тестирования
* Тест кейсы
* Матрица соответствий

## Функциональное тестирование сайта

Функциональное тестирование направлено на то, чтобы каждая функция веб-сайта работала в соответствии с требованиями спецификации. Тестирование функциональности веб-сайта показывает **«Что делает система».**

Попробуем создать чек-лист для тестирования функциональности веб-сайта.

**Тестирование ссылок**

Вы должны проверить:

* Исходящие ссылки
* Корректность внутренних ссылок
* Отстутствие ссылок, ведущих к одной странице
* Ссылки, которые используются для отправки электронной почты админам сайта
* Есть ли страницы, на которые не указаны ссылки
* Отсутствие неработающих ссылок

**Тестирование форм для всех страниц**

Вы используете формы для интерактивного общения с вашими клиентами. Итак, необходимо проверить следующие моменты:

* Действительность входных данных
* Допустимые значения для поля данных
* Недопустимые входные значения для поля данных
* Параметры форм, в которых возможно удаление или любая другая модификация данных.

**Тестирование cookies**

Cookies представляют собой небольшие файлы, которые хранятся на компьютере пользователя после посещения веб-страницы.

* Проверьте сайт с отключенными cookies
* Проверьте сайт с включенными cookies
* Убедитесь, что файлы cookies зашифрованы перед записью на компьютер пользователя
* Проверьте аспекты безопасности при удалении файлов cookies.
* Если cookies имеют продолжительность действия, то следует проверить, активны ли они в указанный период времени.

**HTML / CSS валидация**

* Синтаксические ошибки HTML
* Убедитесь, что сайт доступен для поисковых машин.
* Убедитесь, что ваша веб-страница имеет точную карту сайта в формате XML и HTML

Полезные инструменты для проведения функционального тестирования: **[Selenium](http://www.seleniumhq.org/)**, **[Linux Test Project](https://linux-test-project.github.io/), [JUnit,](http://junit.org/junit4/)**[**Sprinter by Hewlett Packard Entreprise**](https://saas.hpe.com/ru-ru/software/sprinter-manual-software-testing) (ручное тестирование), **[Browserstack](http://www.browserstack.com/)** (ручное и автоматизированное тестирование), **[Usersnap](https://usersnap.com/)** (ручное тестирование).

Перейдя по этой ссылке, Вы сможете прочитать больше о функциональном тестировании - <https://geteasyqa.com/qa/software-testing-types/>

## Usability тестирование сайта (тестирование удобства использования)

Usability тестирование предназначено для оценки вашей веб-страницы с точки зрения конечного пользователя. Это помогает определить соответствие продукта ожиданиям пользователей, выявляет проблемные места в интерфейсе.

**Навигационное тестирование сайта** содержит следующие проверки:

* Все страницы сайта понятны и просты в использовании.
* Кнопки, формы и поля удобны для использования.
* Доступ к главному меню осуществляется со всех страниц.

Чек-лист **тестирования контента**:

* Отсутствуют грамматические, орфографические ошибки
* Изображения имеют соответствующие размеры и размещены правильно
* Проверьте оптимизацию цветовой палитры сайта и размеры шрифтов
* Контент должен быть информативным, понятным, структурированным и логически связанным
* Инструкции ясны и содержат правильную информацию

Наконец, чтобы оценить **удобство использования** вашего веб-портала, просто ответьте на эти вопросы:

* Является ли ваш сайт понятным и удобным?
* Удобна ли навигация?
* Какое впечатление он производит на пользователя?
* Есть лишние или ненужные вещи?

Полезные инструменты для usability тестирования: **[User Zoom](https://www.userzoom.com/), [Reflector,](https://reflector.en.softonic.com/)** [**Loop11**](https://www.loop11.com/)**.**

Больше информации о usability тестировании здесь - <https://geteasyqa.com/qa/software-testing-types/>

## Тестирование UI (интерфейса пользователя)

Тестирование интерфейса пользователя (UI) выполняется для проверки соответствия графического пользовательского интерфейса вашего сайта спецификациям.

Вот некоторые проверки для тестирования интерфейса веб-сайта:

* Соответствие стандартам графических интерфейсов
* Оценка элементов дизайна: макет, цвета, шрифты, размеры шрифтов, ярлыки, текстовые поля, форматирование текста, титры, кнопки, списки, значки, ссылки
* Тестирование с различными разрешениями экрана
* Тестирование локализированных версий: точность перевода, проверка длины имен элементов интерфейса и т. п.
* Тестирование графического интерфейса пользователя на целевых устройствах: смартфоны и планшеты.

Полезные инструменты для UI тестирования: **[FitNesse](http://www.fitnesse.org/), [iMacros,](https://imacros.net/download/)** **[Coded UI,](https://msdn.microsoft.com/en-us/library/dd286726.aspx)** **[Jubula,](https://www.eclipse.org/jubula/)** **[LoadUI](https://www.loadui.org/).**

Больше информации о UI тестировании здесь - <https://geteasyqa.com/qa/software-testing-types/>

## Тестирование совместимости (конфигурационное тестирование)

Тестирование совместимости выполняется для проверки работы сайта при различных программных и аппаратных конфигурациях:

* Конфигурация операционной системы
* Конфигурация браузера
* Конфигурация базы данных

**Кроссплатформенное тестирование сайта** позволяет оценивать работу вашего сайта при разных ОС (как десктопных, так и мобильных): Windows, iOS / Mac OS, Linux, Android, BlackBerry и т. д.

**Кросс-браузерное тестирование сайта** помогает проверить правильность работы сайта в разных конфигурациях браузера: Mozilla Firefox, Google Chrome, Internet Explorer, Opera и т. п.

**Тестирование баз данных** выполняется для обеспечения правильной работы вашего сайта при разных конфигурациях базы данных: Oracle, DB2, MySql, MSSQL Server, Sybase и т.д.

**Совместимость опций печати** также следует упомянуть в плане тестирования вашего веб-сайта:

* Проверьте правильность печати шрифтов, графических изображений страниц, выравнивания страниц
* Убедитесь, что все страницы вашего сайта соответствуют размеру бумаги, определенному в опциях печати.

Вы можете использовать такие инструменты как **[BrowserStack,](https://www.browserstack.com/)** **[CrossBrowserTesting by Smart Bear](https://crossbrowsertesting.com/), [Litmus](https://litmus.com/)**[**, Browsera**](https://www.browsera.com/)**, [Rational Clearcase by IBM](https://www-03.ibm.com/software/products/ru/clearcase),  [Ghostlab](https://vanamco.com/ghostlab/)**для тестирования совместимости сайта.

По этому адресу Вы найдёте больше информации о конфигурационном тестировании - <https://geteasyqa.com/qa/software-testing-types/>

## Тестирование производительности

Тестирование производительности предназначено для определения того, как система работает с точки зрения оперативности и стабильности при определенной нагрузке. Сайт должен выдерживать высокие нагрузки. Методы тестирования производительности веб-сайта содержат:

* Тестирование поведения сайта на уровне или за пределами его ожидаемой рабочей нагрузки**(стресс-тестирование)**
* Тестирование поведения сайта при увеличении рабочей нагрузки**(нагрузочное тестирование)**
* Тестирование способности работать в течение или чуть дольше приемлемого периода **(тестирование стабильности)**
* Тестирование производительности веб-сайта за счет увеличения объема данных в базе данных **(объёмное тестирование)**
* Тестирование производительности веб-сайта, при одновременной логинизации большого количества пользователей**(тестирование параллелизма)**
* Тестирование поведения сайта при непрерывной дополнительной нагрузке **(тестирование на выносливость)**
* Тестирование скорости загрузки страницы

Полезные инструменты для тестирования производительности: **[Apache JMeter](https://jmeter.apache.org/),**[**HP LoadRunner**](https://saas.hpe.com/en-us/software/loadrunner)**, [Silk Performer from Micro Focus](https://www.microfocus.com/products/silk-portfolio/silk-performer/), [WebLOAD](https://www.radview.com/),** **[Gatling](http://gatling.io/).**

Хотите знать больше о тестировании производительности и инструментах, которые используются для этого? Перейдите по этой ссылке <https://geteasyqa.com/ru/blog/best-tools-load-testing>

## Тестирование безопасности

Тестирование безопасности выполняется для проверки системы защиты данных и поддержки функционала.

Вы можете имитировать атаку вредоносного источника, чтобы оценить уровень безопасности вашего сайта (**тестирование проникновения**).

Другой тип тестирования безопасности, **тестирование уязвимости**, позволяет оценить общий объем вовлеченных рисков.

Некоторые проверки для тестирования безопасности:

* Обеспечить невозможность несанкционированного доступа к защищенным страницам
* Автоматическое прекращение проверки сеансов после длительного простоя пользователя
* Тестирование функций безопасности SSL
* Все попытки взлома, сообщения об ошибках и т. п. должны регистрироваться и сохраняться в отдельном файле для дальнейшего анализа.
* Проверьте работу captcha с помощью автоматических скриптов
* Убедитесь, что файлы с ограниченным доступом не загружаются без соответствующего разрешения
* Убедитесь, что при вводе неправильного пароля или имени пользователя нет возможности входа в систему

Полезные инструменты для тестирования безопасности сайта: **[Retina CS Community](https://www.beyondtrust.com/products/retina-cs-community/),**[**OWASP Zed Attack Proxy**](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)**, [Veracode,](https://www.veracode.com/)** **[Google Nogotofail,](https://github.com/google/nogotofail)** [**SQL Map**](http://sqlmap.org/)**.**

Больше информации о тестировании безопасности Вы сможете найти тут - <https://geteasyqa.com/qa/software-testing-types/>

## Тестирование, связанное с изменениями

Тестирование, связанное с изменениями, имеет две основные цели:

* Убедиться в том, что все обнаруженные баги действительно успешно исправлены (**повторное тестирование или подтверждающее тестирование**). Проще говоря, вы должны запустить тест кейсы, с изначально обнаруженными багами, и на этот раз они исполняются без каких-либо проблем.
* Убедиться в том, что не возникли новые баги изменений (**регрессионное тестирование**). Помимо тест кейсов с обнаруженными багами, оно также содержит тест кейсы, проверяющие все функциональные возможности вашего сайта.

[**Selenium**](http://www.seleniumhq.org/)**,**[**HP Quick Test Professional,**](https://saas.hpe.com/en-us/software/uft)[**TestComplete**](https://smartbear.com/product/testcomplete/overview/)**, [TestDrive](https://www.origsoft.com/products/testdrive/), [SoapUI](http://www.soapui.org/)**часто используются для тестирования, связанного с изменениями.

Здесь Вы сможете найти больше информации о тестировании, связанном с изменениями -<https://geteasyqa.com/qa/software-testing-types/>

## Тестирование мобильной версии сайта

Как вы могли заметить, некоторые из проверок были связаны с мобильной версией вашего сайта. В настоящее время число людей, использующих только мобильные устройства для доступа в Интернет, имеет тенденцию к стабильному росту. Вот почему, очень важно адаптировать версию сайта к мобильным устройствам.

Вот несколько советов для того, чтобы сделать эффективным тестирование сайта на мобильных устройствах:

* Проверьте совместимость со смартфонами и планшетами
* Убедитесь, что навигация по сайту максимально проста
* Оптимизируйте время загрузки вашего сайта
* Убедитесь, что кнопки имеют достаточный размер для людей с большим пальцем
* Оптимизируйте размер всех изображений
* Не используйте Flash и всплывающие окна
* Используйте маркеры и короткие предложения
* Убедитесь, что ваш номер телефона может быть набран с помощью одного клика
* Убедитесь, что веб-сайт может получить доступ к вашему местоположению через GPS

Полезные инструменты для тестирования мобильной версии сайта - **[BrowserStack](https://www.browserstack.com/), [Perfecto Mobile Continuous Quality Lab](https://www.perfectomobile.com/continuous-quality-lab), [Windows Phone Emulator](https://msdn.microsoft.com/en-us/library/windows/apps/ff402563), [Android Studio emulator](https://developer.android.com/studio/index.html), Google's Mobile-Friendly Test, [Google’s Page Speed Online](https://developers.google.com/speed/pagespeed/insights/)**.

Узнайте больше о мобильном тестировании и его инструментах-<https://geteasyqa.com/qa/test-website-mobile/>

## Бета-тестирование

Бета-тестирование - заключительная предварительная стадия тестирования. Как правило, это делают конечные пользователи, которые не являются сотрудниками компании.

При бета-тестировании ваш сайт попадает в руки реальных пользователей, чтобы обнаружить любые недостатки с их точки зрения, которые вы не хотели бы иметь в своей окончательной, релизной версии.

Такие инструменты, как **[HockeyApp](https://hockeyapp.net/%22%20%5Cl%20%22s), [Ubertesters](https://ubertesters.com/),**и[**TestFlight**](https://developer.apple.com/testflight/), являются всемирно используемыми платформами для бета-тестирования.

Теперь, когда мы рассмотрели основные этапы процесса тестирования веб-сайта, давайте попробуем найти баг и сообщить об этом с помощью одного из реальных инструментов веб-тестирования, расширения для браузера EasyQA Chrome Extension.

## Как проводить тестирование сайта с помощью EasyQA Chrome Extension

EasyQA Chrome Extension позволяет создать баг репорт с вашего веб-сайта или веб-приложения, не тратя время на отправку информации,помогает в кратчайший срок начать работу по исправлению бага.

Использовать EasyQA Chrome Extension для работы с багами очень просто. Всё, что вам нужно сделать, это:

* Создайте токен для Вашего Проекта
* Установите EasyQA Chrome Extension в свой браузер
* Залогиньтесь (по желанию).



Основные возможности EasyQA Chrome Extension:

* Возможность репортить баг непосредственно с вашего веб-сайта или веб-приложения
* Возможность создания баг репорта как для зарегистрированных, так и для незарегистрированных пользователей
* Создание скриншотов
* Видеозапись обнаруженных багов
* Подготовка и отправка баг репорта в иллюстрированной форме



Перейдя по этой ссылке <https://geteasyqa.com/ru/sdk/extension-for-chrome/>, вы сможете найти полное руководство по установке и использованию EasyQA Chrome Extension.