**AJAX**, Ajax ([ˈeɪdʒæks](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D0%B4%D1%83%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D1%84%D0%BE%D0%BD%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82), от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Asynchronous Javascript and XML* — «асинхронный [JavaScript](https://ru.wikipedia.org/wiki/JavaScript" \o "JavaScript) и [XML](https://ru.wikipedia.org/wiki/XML)») — подход к построению интерактивных [пользовательских интерфейсов](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%B8%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81) [веб-приложений](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5), заключающийся в «фоновом» [обмене данными](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BC%D0%B5%D0%BD_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%BC%D0%B8) [браузера](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B1%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80) с [веб-сервером](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80). В результате при обновлении данных [веб-страница](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0) не перезагружается полностью, и веб-приложения становятся быстрее и удобнее.

## История

Впервые термин AJAX был публично использован [18 февраля](https://ru.wikipedia.org/wiki/18_%D1%84%D0%B5%D0%B2%D1%80%D0%B0%D0%BB%D1%8F) [2005 года](https://ru.wikipedia.org/wiki/2005_%D0%B3%D0%BE%D0%B4) в статье Джесси Джеймса Гарретта (*Jesse James Garrett*) «Новый подход к веб-приложениям». Гарретт придумал термин, когда ему пришлось как-то назвать новый набор технологий, предлагаемый им клиенту.

Однако в той или иной форме многие технологии были доступны и использовались гораздо раньше, например в подходе «Remote Scripting», предложенном компанией [Microsoft](https://ru.wikipedia.org/wiki/Microsoft) в [1998 году](https://ru.wikipedia.org/wiki/1998_%D0%B3%D0%BE%D0%B4), или с использованием [HTML](https://ru.wikipedia.org/wiki/HTML)-элемента IFRAME, появившегося в [Internet Explorer](https://ru.wikipedia.org/wiki/Internet_Explorer" \o "Internet Explorer) 3 в [1996 году](https://ru.wikipedia.org/wiki/1996_%D0%B3%D0%BE%D0%B4).

AJAX стал особенно популярен после использования его компанией [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BA%D0%BE%D0%BC%D0%BF%D0%B0%D0%BD%D0%B8%D1%8F)" \o "Google (компания)) в сервисах [Gmail](https://ru.wikipedia.org/wiki/Gmail" \o "Gmail), [Google Maps](https://ru.wikipedia.org/wiki/Google_Maps" \o "Google Maps) и [Google Suggest](https://ru.wikipedia.org/wiki/Google_Suggest" \o "Google Suggest).

## Преимущества

**Экономия трафика**

Использование AJAX позволяет [сократить трафик](https://ru.wikipedia.org/w/index.php?title=%D0%AD%D0%BA%D0%BE%D0%BD%D0%BE%D0%BC%D0%B8%D1%8F_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0&action=edit&redlink=1) при работе с веб-приложением благодаря тому, что вместо загрузки всей страницы достаточно загрузить только изменившуюся часть или вообще только получить/передать набор данных в формате [JSON](https://ru.wikipedia.org/wiki/JSON) или [XML](https://ru.wikipedia.org/wiki/XML), а затем изменить содержимое страницы с помощью JavaScript.

**Уменьшение нагрузки на сервер**

При правильной реализации AJAX позволяет снизить нагрузку на сервер в несколько раз.

В частности, все страницы сайта чаще всего генерируются по одному шаблону, включая неизменные элементы («шапка», «навигационная панель», «подвал» и т. д.), для генерации которых требуются обращения к разным файлам, время на обработку скриптов (а иногда и запросы к БД) — всё это можно опустить, если заменить полную загрузку страницы генерацией и передачей лишь содержательной части.

**Ускорение реакции интерфейса**

Поскольку загрузка изменившейся части значительно быстрее, то пользователь видит результат своих действий быстрее и без мерцания страницы (возникающего при полной перезагрузке).

**Возможности для интерактивной обработки**

Например, при вводе поискового запроса в [Google](https://ru.wikipedia.org/wiki/Google_(%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)" \o "Google (поисковая система)) выводится подсказка с возможными вариантами запроса. На многих сайтах при регистрации пользователь вводит имя, и сразу же видит, доступно это имя или нет. AJAX удобен для программирования [чатов](https://ru.wikipedia.org/wiki/%D0%A7%D0%B0%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0)), [административных панелей](https://ru.wikipedia.org/w/index.php?title=%D0%90%D0%B4%D0%BC%D0%B8%D0%BD%D0%B8%D1%81%D1%82%D1%80%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D0%BD%D0%B5%D0%BB%D1%8C&action=edit&redlink=1) и других инструментов, которые выводят меняющиеся со временем данные.

[**Мультимедиа**](https://ru.wikipedia.org/wiki/%D0%9C%D1%83%D0%BB%D1%8C%D1%82%D0%B8%D0%BC%D0%B5%D0%B4%D0%B8%D0%B0)**не останавливается**

Страница не перезагружается, плеер продолжает работать. Потому AJAX ценен на аудио- и видеохостингах.

## Недостатки

**Отсутствие интеграции со стандартными инструментами браузера**

Динамически создаваемые страницы не регистрируются браузером в истории посещения страниц, поэтому не работает кнопка «Назад», предоставляющая пользователям возможность вернуться к просмотренным ранее страницам, но существуют скрипты, которые могут решить эту проблему.

Другой недостаток изменения содержимого страницы при постоянном [URL](https://ru.wikipedia.org/wiki/URL) заключается в невозможности сохранения закладки на желаемый материал. Проблему можно успешно решить с помощью History.pushState[[4]](https://ru.wikipedia.org/wiki/AJAX" \l "cite_note-4).

**Динамически загружаемое содержимое недоступно поисковикам (если не проверять запрос, обычный он или [XMLHttpRequest](https://ru.wikipedia.org/wiki/XMLHttpRequest" \o "XMLHttpRequest))**

Поисковые машины не могут выполнять [JavaScript](https://ru.wikipedia.org/wiki/JavaScript" \o "JavaScript), поэтому разработчики должны позаботиться об альтернативных способах доступа к содержимому сайта (но на 2017 год уже не актуально)

**Старые методы учёта статистики сайтов становятся неактуальными**

Многие сервисы статистики ведут учёт просмотров новых страниц сайта. Для сайтов, страницы которых широко используют AJAX, такая статистика теряет актуальность.

**Усложнение проекта**

Перераспределяется логика обработки данных — происходит выделение и частичный перенос на сторону клиента процессов первичного форматирования данных. Это усложняет контроль целостности форматов и типов. Конечный эффект технологии может быть нивелирован необоснованным ростом затрат на кодирование и управление проектом, а также риском снижения доступности сервиса для конечных пользователей.

**Требуется включённый JavaScript в браузере**

JavaScript может быть выключен из соображений безопасности. И, конечно же, AJAX-страницы труднодоступны неполнофункциональным браузерам, роботам и [веб-архивам](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B0%D1%80%D1%85%D0%B8%D0%B2).

**Проблемы с отображением нестандартных кодировок в некоторых сценариях ajax-скриптов**

О проблемах AJAX и кириллицы много сказано в обсуждениях в Интернете[[5]](https://ru.wikipedia.org/wiki/AJAX#cite_note-5).

**Низкая скорость при грубом программировании**

Казалось бы, AJAX предназначен именно для повышения скорости. Но когда AJAX-запросов на одной странице много и, например, по каждому щелчку подгружается список, AJAX-страница становится даже медленнее традиционной.

**Плохое поведение на ненадёжных соединениях**

Если связь часто теряется (из-за потери несущей или перегрузки канала), обычную страницу можно как минимум перезагрузить. AJAX-страницу (например, с «бесконечной» прокруткой) приходится перезагружать с самого начала и искать, где остановился. Параллельная работа — особенность AJAX — здесь оказывает медвежью услугу, деля и без того узкий канал на множество маленьких соединений, и велика вероятность, что какое-то будет разорвано. Частично решается API истории.

**Риск**[**фабрикации запросов другими сайтами**](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%B4%D0%B5%D0%BB%D0%BA%D0%B0_%D0%BC%D0%B5%D0%B6%D1%81%D0%B0%D0%B9%D1%82%D0%BE%D0%B2%D1%8B%D1%85_%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81%D0%BE%D0%B2)

Результат работы AJAX-запроса может являться [JavaScript](https://ru.wikipedia.org/wiki/JavaScript" \o "JavaScript)-кодом (в частности, [JSON](https://ru.wikipedia.org/wiki/JSON)). [XMLHttpRequest](https://ru.wikipedia.org/wiki/XMLHttpRequest" \o "XMLHttpRequest) действует только [в пределах одного домена](https://ru.wikipedia.org/wiki/Same_Domain_Policy), а вот тег <script> — нет.

## Безопасность

Hijacking – главная проблема AJAX технологии, так как в основе AJAX лежит обмен сообщениями между сервером и клиентом посредством JavaScript-сообщений. Hijacking нацелен на перехват таких сообщений. Традиционный обмен сообщениями между веб-клиентом и сервером лучше защищен от подобных атак благодаря технологии SOP (Same Origin Policy). Какой-либо защиты сообщений, формируемых с помощью JavaScript, пока не наблюдается.

Способы защиты:

* сделать невозможным прямое  
  выполнение JS-ответа
* нужно организовать отсылку в известном  
  направлении всех кривых или просто подозрительных запросов

## Будущее технологии

Сейчас технология имеет очень активное развитие и не собирается сдавать свои позиции.