

Cookie. Session.

Понятие сервисно-ориентированной архитектуры (SOA).

API для взаимодействия программных систем через Web.

SOAP.

REST.

Идея HTTP cookie в протоколе HTTP

- Идея cookie – отправлять Web сервером небольшой фрагмент данных для сохранения на компьютере пользователя, который автоматически возвращается браузером при повторном обращении клиента к данному серверу
- Некоторые из типичных применений cookie :
 - аутентификация пользователей
 - хранение персональных предпочтений и настроек
 - отслеживание состояния сеанса доступа пользователя
 - ведение статистики о пользователе

Назначение и особенности HTTP cookie

- Спецификации: RFC2109 И RFC 2965 (IETF)
- Минимальные объемы: 300 порций по 4096 байт
20 cookie для одного сервера или домена
- Есть ограничения по браузерам !
- Разные сроки устаревания: cookie сеанса и постоянные cookie (с определенной датой)
- Кто добавляет cookie?

Схема работы HTTP cookie

- Запрос браузера веб-серверу:
GET /index.html HTTP/1.1
Host: www.web-server.ru
- Ответ веб-сервера с установкой cookie
HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: name=value
- Запрос с возвратом cookie при повторном обращении к серверу
GET /спец.html HTTP/1.1
Host: www.web-server.ru
Cookie: name=value
...

Заголовок HTTP-ответа Set-Cookie

- Установка сервером cookie через заголовок Set-Cookie:
Set-Cookie: <имя>=<значение>; expires=<дата>; path=<путь>; domain=<домен>; secure
- имя: строка без использования пробела, точки с запятой, ...
- значение: строка без использования пробела, точки с запятой, ...
- expires=<дата> - дата истечения срока действия cookie
например, Wednesday, 01-Dec-2022 00:00:00 GMT
- domain=<домен> - диапазон доменов
- path=<путь> - устанавливает подмножество документов, на которые распространяется действие cookie
- secure: флаг, требующий возвращения cookie только при условии защищенного соединения

Заголовок запроса HTTP cookie

- Сообщение о наличии на клиенте cookie, определенных для данной страницы осуществляется через заголовок запроса Cookie:

Cookie: <имя>=<значение>[; <имя>=<значение>]

- имена в парах могут повторяться;
- в начале размещаются наиболее точно специфицированные cookie;
- значение: строка без использования пробела, точки с запятой.

Установка HTTP cookie через HTML

Вспомним мета-тег заголовка.

Он пригоден и для установки cookie :

```
<META http-equiv="Set-Cookie"  
content="NAME=value; EXPIRES=date;  
DOMAIN=domain_name; PATH=path; SECURE" />
```

Session: настройка состояния сеанса

Пакет Microsoft.AspNetCore.Session:

- неявно включается платформой.
- Предоставляет ПО промежуточного слоя для управления состоянием сеанса.

Чтобы включить сеанс ПО промежуточного слоя для сеансов, Startup должен содержать:

- Вызов к AddSession в ConfigureServices.
- Вызов к UseSession в Configure.

Service-oriented architecture (SOA).

Сервис(но)-ориентированная архитектура.

Под сервис-ориентированной архитектурой понимается **модульный** подход к проектированию прикладных информационных систем, который руководствуется следующими принципами:

- явное отделение бизнес-логики прикладной системы от логики представления информации;
- реализация бизнес-логики прикладной системы в виде программных модулей (сервисов), которые доступны извне (пользователям и другим модулям), чаще всего в режиме "запрос-ответ", через четко определенные формальные интерфейсы доступа;
- при этом "потребитель сервиса" (прикладная система или другой сервис), имеет возможность вызвать сервис через программные интерфейсы, используя соответствующие коммуникационные механизмы.

Базовыми понятиями в такой архитектуре являются *"информационная услуга"* (сервис) и *"композиционное приложение"*.

SOA: информационная услуга (сервис).

Информационная услуга (сервис) -

это атомарная прикладная функция автоматизированной системы, которая пригодна для использования при разработке приложений, реализующих прикладную логику автоматизируемых процессов как в самой системе, так и для использования в приложениях других автоматизированных систем.

Сервис обычно характеризуется следующими свойствами:

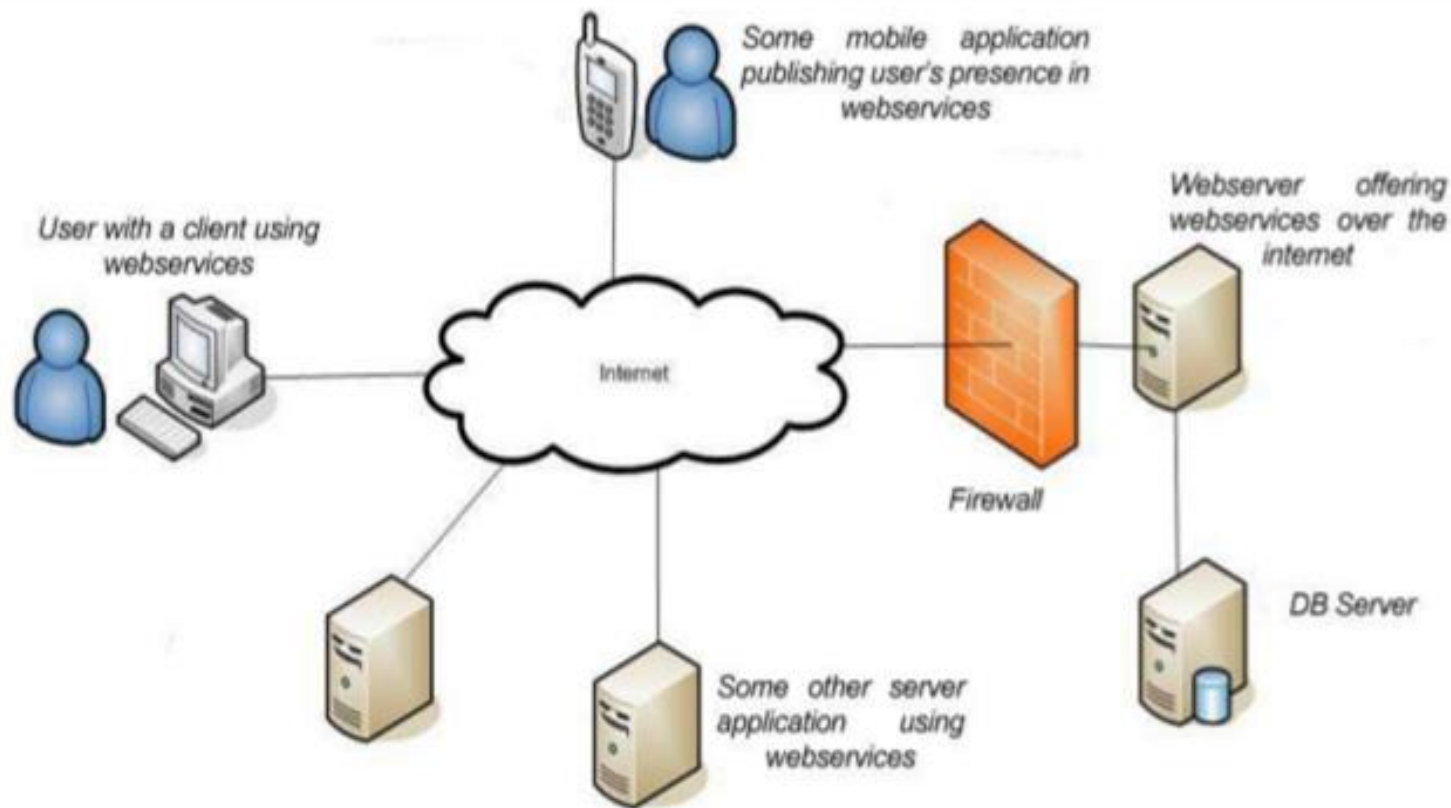
- возможность многократного применения;
- услуга может быть определена одним или несколькими технологически независимыми интерфейсами;
- выделенные услуги слабо связаны между собой, и каждая из них может быть вызвана посредством *коммуникационных протоколов*, обеспечивающих возможность взаимодействия услуг между собой.

SOA: композитное приложение

Композитное (составное) приложение - программное решение для конкретной прикладной проблемы, которое связывает прикладную логику процесса с источниками данных и информационных услуг, хранящихся на *гетерогенном* множестве базовых информационных систем.

Обычно композитные приложения ассоциированы с процессами деятельности и могут объединять различные этапы процессов, представляя их пользователю через единый *интерфейс*.

Необходимость API



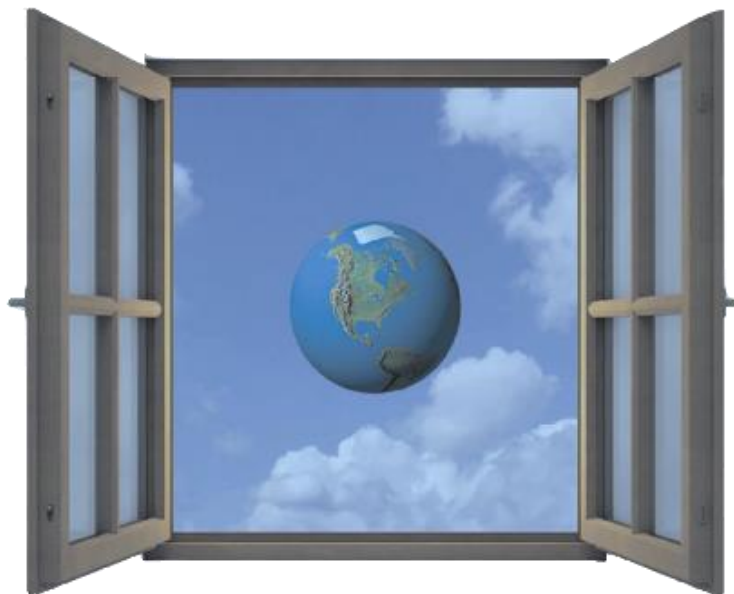
Web service (Веб-сервис, веб-служба) – сервис (набор методов), предоставляемый приложением и доступный для использования по сети:

- Стандартизированный способ для взаимодействия разнородных приложений;
- Без ограничений на ОС, язык программирования или устройство;
- Предоставление услуг для любого приложения.

Web API = API через Web

API (Application Programming Interface – программный интерфейс приложения) — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой. Обычно входит в описание какого-либо интернет-протокола (например, RPC), программного каркаса (фреймворка) или стандарта вызовов функций операционной системы. Часто реализуется отдельной программной библиотекой или сервисом операционной системы. Используется программистами при написании всевозможных приложений.

API – окно в мир (ключ к автоматизированному использованию вашего сервиса)



Веб-сервис

Веб-сервис – идентифицируемая веб-адресом (URL) программная система со стандартизированными интерфейсами.

Веб-службы могут взаимодействовать между собой и со сторонними приложениями посредством сообщений, основанных на определенных протоколах.

Интерфейс сервиса описан в Web Service Description Language (WSDL)

Веб-сервис – единица модульности при создании SOA-приложения.

Характеристики:

- функциональная совместимость;
- расширяемость;
- возможность машинной обработки описания.

Веб-сервисы взаимодействуют между собой через протокол SOAP в соответствии с WSDL.

SOAP = Simple Object Access Protocol

SOAP (Simple Object Access Protocol – простой протокол доступа к объектам) - протокол обмена структурированными сообщениями в распределенной вычислительной среде; первоначально предназначался для удаленного вызова процедур (RPC), сейчас используется так же для обмена произвольными сообщениями в формате XML.

SOAP может использоваться с любым протоколом прикладного уровня: SMTP, FTP, HTTP, HTTPS и др.

Как правило, применяется в бизнес-приложениях и на основе ранее существовавших систем.

Ориентирован на написание методов и ОО-подход.

REST

REST (Representational State Transfer - «передача состояния представления») - архитектурный стиль взаимодействия компонентов распределённого приложения в сети (термин введен Roy Thomas Fielding). REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой системы, в определённых случаях приводящих к повышению производительности и упрощению архитектуры.

В широком смысле компоненты в REST взаимодействуют наподобие взаимодействия клиентов и серверов в WWW.

REST является альтернативой другим вариантам RPC (SOAP, COM+, etc...).

REST: требования к архитектуре

- Модель клиент-сервер.
- Отсутствие состояния.
- Кэширование.
- Единообразие интерфейса.
- Слои.
- Код по требованию (необязательное ограничение).

Системы, поддерживающие REST, называются RESTful-системами:

- Явное использование HTTP-методов.
- Предоставление URI, аналогичных структуре каталогов.
- Передача данных в XML, JavaScript Object Notation (JSON).

CRUD = Create, Read, Update, Delete

CRUD – аббревиатура, обозначающая четыре базовые функции, используемые при работе с базами данных: создание (create), чтение (read), модификация (update), удаление (delete). Термин введен James Martin в 1983 г. как стандартная классификация функций по манипуляции данными.

В системах, реализующих доступ к базе данных через API в стиле REST, эти функции реализуются зачастую (но не обязательно) через HTTP-методы POST, GET, PUT и DELETE соответственно:

Действие -> HTTP-метод

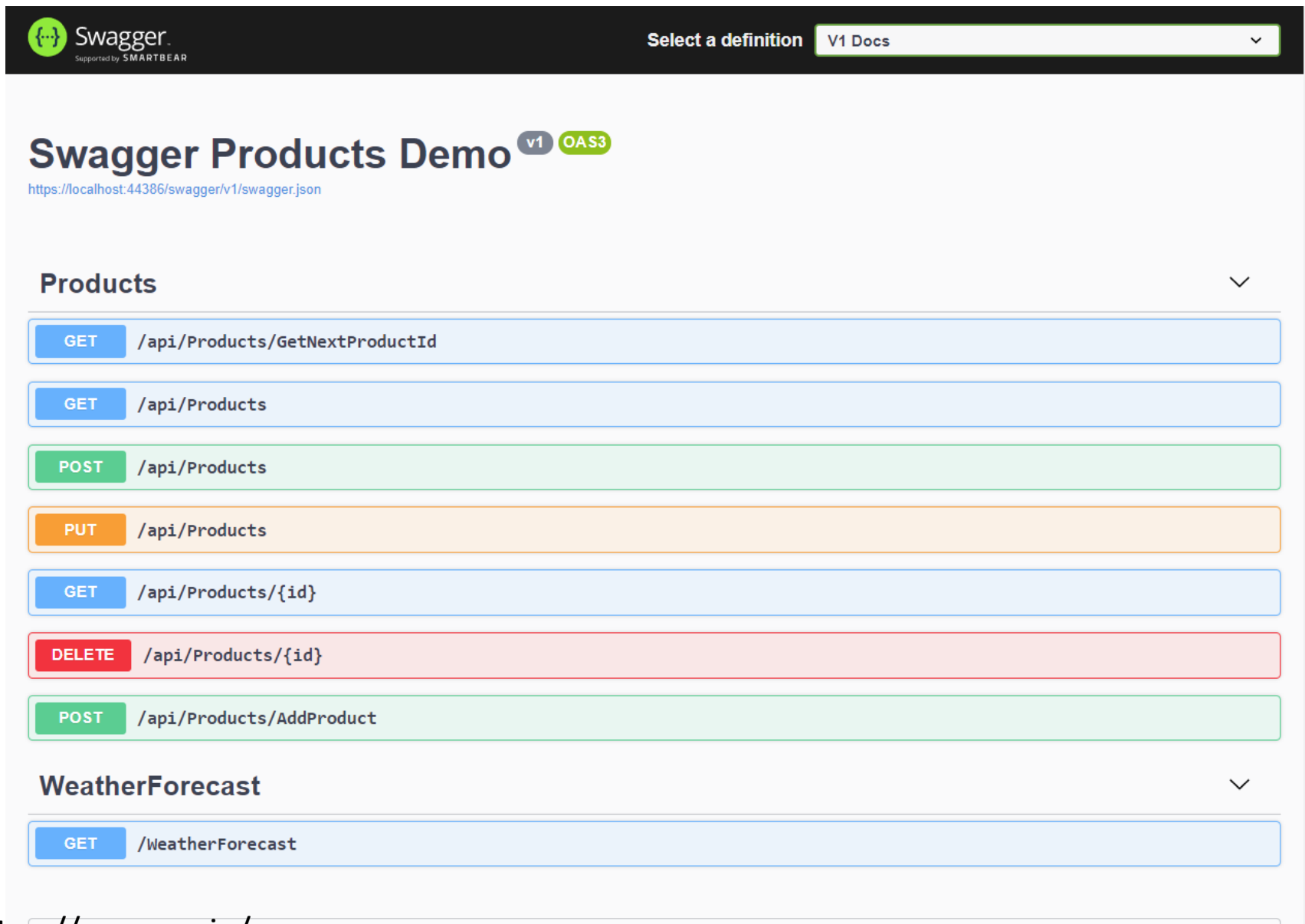
Create	-> POST
Read	-> GET
Update	-> PUT
Delete	-> DELETE

<https://habr.com/ru/post/483202/>
<https://metanit.com/sharp/mvc/12.1.php>

REST: принципы построения адресов

URI	HTTP-метод	Действие
/products/	GET	получить список всех товаров
/products/3	GET	получить товар с id=3
/products/	PUT	изменить товар (данные в теле запроса)
/products/	POST	добавить товар (данные в теле запроса)
/products/4	DELETE	удалить товар

REST: использование Swagger



The screenshot displays the Swagger UI interface for a service named "Swagger Products Demo". At the top, the Swagger logo is visible, along with the text "Supported by SMARTBEAR". A dropdown menu labeled "Select a definition" is set to "V1 Docs". Below the header, the title "Swagger Products Demo" is shown with "v1" and "OAS3" tags, and the URL "https://localhost:44386/swagger/v1/swagger.json". The main content is organized into two sections: "Products" and "WeatherForecast". The "Products" section lists seven endpoints: a GET endpoint for "/api/Products/GetNextProductId", a GET endpoint for "/api/Products", a POST endpoint for "/api/Products", a PUT endpoint for "/api/Products", a GET endpoint for "/api/Products/{id}", a DELETE endpoint for "/api/Products/{id}", and a POST endpoint for "/api/Products/AddProduct". The "WeatherForecast" section lists a single GET endpoint for "/WeatherForecast". Each endpoint is represented by a colored bar indicating its HTTP method.

Swagger

Supported by SMARTBEAR

Select a definition V1 Docs

Swagger Products Demo v1 OAS3

<https://localhost:44386/swagger/v1/swagger.json>

Products

- GET /api/Products/GetNextProductId
- GET /api/Products
- POST /api/Products
- PUT /api/Products
- GET /api/Products/{id}
- DELETE /api/Products/{id}
- POST /api/Products/AddProduct

WeatherForecast

- GET /WeatherForecast

SOAP vs REST

Критерий	SOAP	RESTful
Назначение	Инкапсулирует бизнес-логику	Доступ к ресурсам/данным
Методология разработки	Объектно-ориентированная (сервисно-)	Ресурсно-ориентированная
Независимость от языка	Да	Да
Независимость от платформы	Да	Да
Независимость от транспорта	Да	Нет, только HTTP
Стандартизирован	Да	Нет
Безопасность	SSL, WSL-Security	SSL
Транзакции	WS-AtomicTransaction	Нет
Надежная доставка	WS-ReliableMessaging	Контроль со стороны приложения

SOAP vs REST

Критерий	SOAP	RESTful
Производительность	Ниже	Выше
Кэширование	Нет	Есть для метода GET
Транспорт	HTTP, SMTP	HTTP
Размер сообщений	Большой, служебные данные	Небольшой
Протокол сообщений	XML	XML, JSON, любой тип MIME
Описание сервисов	WSDL	Формального нет
Инструменты разработки	Требуются	Можно без них

Выводы: REST против SOAP можно перефразировать как «Простота против Стандарта». В случае REST (простота) у вас будет скорость, расширяемость и поддержка многих форматов. В случае с SOAP у вас будет больше возможностей по безопасности (WS-security) и транзакционная безопасность (ACID).