**What is CORS? (For Beginners)**

**CORS** (Cross-Origin Resource Sharing) is a security feature implemented by browsers. It allows websites to request data from other websites, but only if the second website gives permission.

Normally, websites are restricted by the **Same-Origin Policy**, meaning a website can only request data from the same domain (or origin) that it was loaded from. **CORS** allows developers to bypass this restriction safely, but only when the server (the second website) says it's okay.

**CORS Example:**

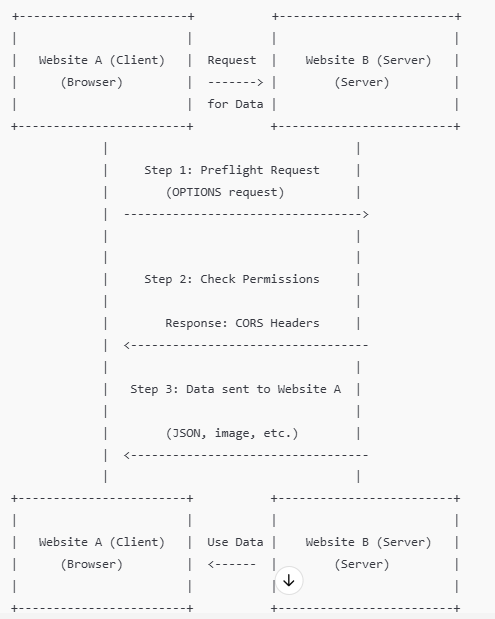
Let’s say **Website A** wants to get some data (like an image or information) from **Website B**, which is on a different domain. Without CORS, the browser would block that request. However, if **Website B** allows it, it will send a special response (CORS header), telling **Website A** it can access the data.

**Simple Steps in CORS:**

1. **Website A** asks **Website B** for data.
2. **Website B** checks if it's allowed for **Website A** to access the data.
3. If allowed, **Website B** sends permission to **Website A**.
4. **Website A** gets the data and uses it.

**Sequence Diagram:**

Here’s a simple **sequence diagram** to show the steps involved in CORS:



**Breakdown of the Diagram:**

1. **Website A (Client)** sends a **request** for data to **Website B (Server)**. This could be an image, text, or other information.
2. If **Website A** is making a request to a different website (domain), the browser first sends a **preflight request** (this is an OPTIONS request) to **Website B**. This checks if **Website A** is allowed to access the data.
3. **Website B** checks its settings (CORS policy) and decides whether to allow **Website A** to access the data. It sends a response with **CORS headers** like Access-Control-Allow-Origin, saying whether or not the request is allowed.
4. If **Website B** allows the request, it sends the actual **data** back to **Website A** (for example, an image or some text).
5. **Website A** receives the data and uses it, like displaying the image or showing the information on the page.

**Simple Example in Action:**

1. **Website A** (like a news website) wants to show weather data from **Website B** (an API).
2. **Website A** asks **Website B** for the data.
3. The browser sends a **preflight request** to check if **Website B** allows this request.
4. **Website B** sends back permission and the weather data.
5. **Website A** receives the data and shows the weather info on its page.

**Key Points to Remember:**

* **CORS** is like asking for permission to access data from another website.
* Without CORS, browsers block requests between different websites for security.
* **CORS headers** are the server's way of saying, "Yes, you're allowed to access this data."

By using CORS, websites can safely share data between different domains.