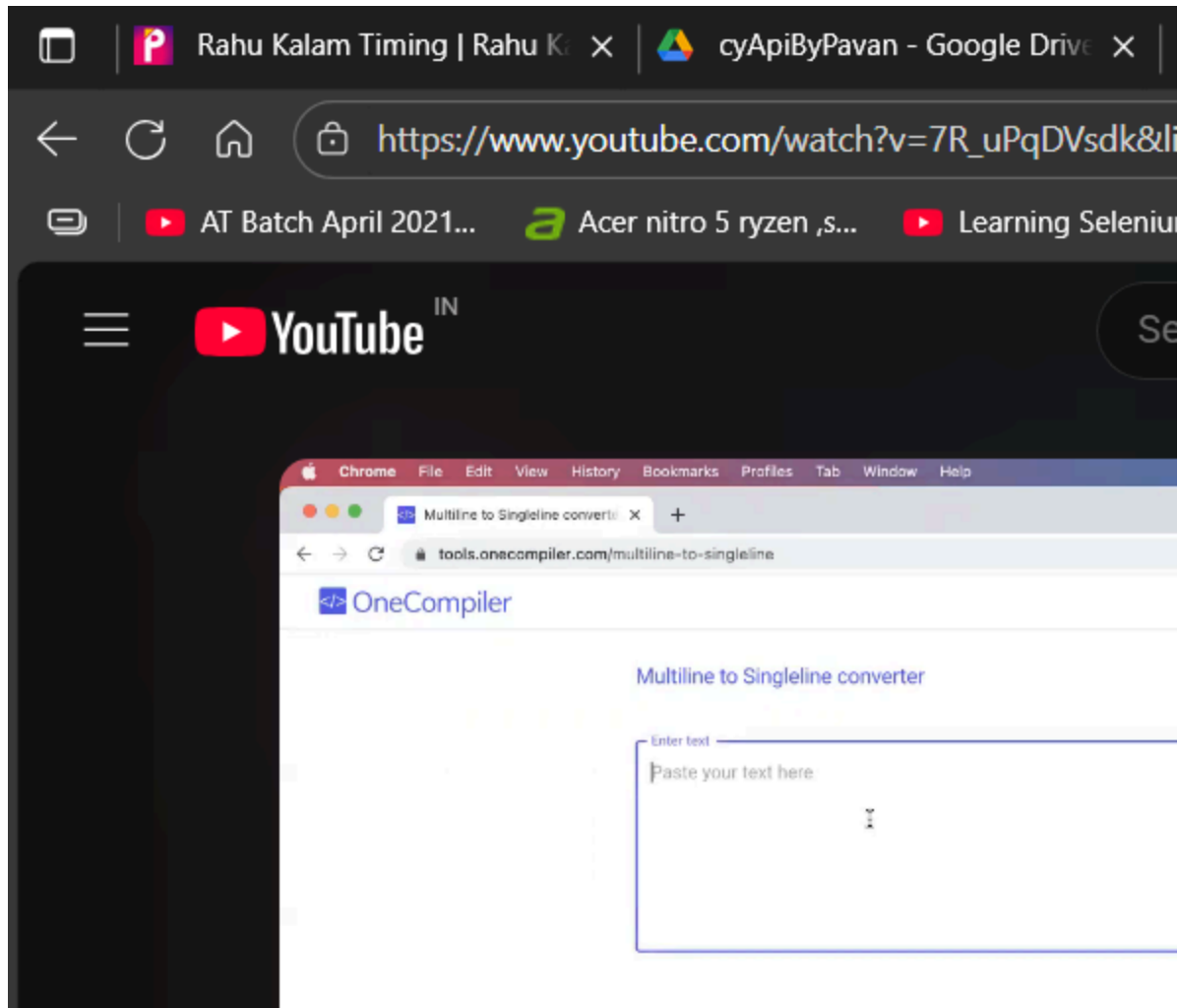


```
1 // Install xml2js library
2 // npm install xml2js
3
4 //import the library.
5 const xml2js = require('xml2js');
6
7 //create instance of xml2js.
8 //we have parser method to parse xml.
9
10 const parser = new xml2js.Parser({ explicitArray: false });
```

Continue below the above lines–



Ocd

YouTube video player interface showing a tutorial titled "API Testing using Cypress | How To Handle XML Payload & Parsing XML Response | Part 6". The video content displays a web browser window with a "Multiline to Singleline converter" tool. The tool has a "Paste your text here" input area and a "Convert" button. Below the input area, it shows "Converted text comes here". The video player includes a sidebar with a list of related videos, a channel name "SDET-QA" with 793K subscribers, and a view count of 10K views 2 years ago. The video player also shows a search bar, a "Create" button, and a "Chat Replay is disabled for this Premiere." message.

API Testing using Cypress | How To Handle XML Payload & Parsing XML Response | Part 6

SDET-QA 793K subscribers

10K views 2 years ago #xml #cypress #apitesting

Chat Replay is disabled for this Premiere.

Mandatory for xml to be in one line else compile error.

```

1 describe("XML Parsing",()=>{
2
3     //xml has to be in one line else we get compile error.
4     //use multi to single line converter.
5
6     const xmlPayload=<Pet>    <id>0</id>    <Category>
7 <id>0</id>    <name>Dog</name>    </Category>
8 <name>Jimmy</name>    <photoUrls>
9 <photoUrl>string</photoUrl>    </photoUrls>
10 <tags>    <Tag>    <id>0</id>
11 <name>string</name>    </Tag>    </tags>
12 <status>available</status> </Pet>"
13     let petid=null;
14
15     before("Creating new PET",()=>{
16         cy.request({
17             method: 'POST',
18             url: 'https://petstore.swagger.io/v2/pet',
19             body:xmlPayload,
20             //content type is for sending request.
21             //accept for telling what type of response it is.
22             headers:{'Content-Type':'application/xml',
23                     'accept':'application/xml'
24             }
25         }).then((response) =>{
26             expect(response.status).to.eq(200);
27             //parse string converts xml to json object.
28             //error if any will be stored in err.
29             //proper response if any will be stored in result.
30             parser.parseString(response.body, (err,result) => {
31                 petid=result.Pet.id;
32
33             })
34         });
35     });

```

codesnap.dev

Write IT block as separate block below the line 35 –

```
1  it("Fetching Pet data-parsing xml response",()=>{
2      cy.request({
3          method: 'GET',
4          url: "https://petstore.swagger.io/v2/pet/"+petid,
5          //since no body, so content type not mandatory to write.
6          headers:{'accept':'application/xml'}
7      })
8      .then((response) =>{
9          expect(response.status).to.eq(200);
10         parser.parseString(response.body, (err,result) => {
11             expect(result.Pet.id).equal(petid);
12             expect(result.Pet.name).equal("Jimmy")
13         })
14     });
15 });
16
17
18 })
```