Contenido

Ajax 2

Método GET 2

En la vista de Contact 2

Crear el path en el archivo urls.py 2

Crear el bloque correspondiente en el archivo base.html 2

En el template contact.html 2

Verificar su funcionamiento 3

Método POST 3

En la vista de Contact 3

Crear el path en el archivo urls.py 3

En el template contact.html 3

Dentro del bloque `contentJs` 3

Verificar su funcionamiento 4

Práctica, utilizando JavaScript 4

Función para obtener cualquier cookie almacenada 4

Función para ejecutar un servicio con parámetros 5

Servicio get_countries 5

Servicio get_cities 6

Método de vista para servicio get_cities 6

Ajax

Significa JavaScript asíncrono y XML (Asynchronous JavaScript and XML). Es un conjunto de técnicas de desarrollo web que permiten que las aplicaciones web funcionen de forma asíncrona, procesando cualquier solicitud al servidor en segundo plano.

Método GET

En la vista de Contact

```
6  from django.http                    import JsonResponse
7  from django.views.decorators.http   import require_http_methods, require_POST
```

```
34  @require_http_methods(['GET'])
35  def get_countries(request):
36      countries = ['México', 'USA', 'Canada']
37      return JsonResponse({'countries':countries})
```

Crear el path en el archivo urls.py

```
7      path('get_countries/', views.get_countries, name='get_countries'),
```

Crear el bloque correspondiente en el archivo base.html

```
112        // Get the current year for the copy
113        $("#year").text(new Date().getFullYe
114    </script>
115    {% block contentJs %} {% endblock %}
116  </body>
```

En el template contact.html

```
 8          <div class="about-heading-content">
 9          {% comment %} Test Ajax {% endcomment %}
10          <a href="#" onclick='get_countries();' class='btn btn-primary'> Obtiene países</a>
11          <select name="country" id="country" class='text-primary' ></select>
```

```
63
64  {% block contentJs %}
65  <script>
66      function get_countries() {
67          $.ajax({
68              url: '/contact/get_countries/',
69              type: 'GET',
70              dataType: 'json',
71              success: function (data) {
72                  var opcionesSelect = $('#country');
73                  $.each(data.countries, function (index, country) {
74                      opcionesSelect.append($('<option>').text(country).val(country));
75                  });
76              }
77          });
78      }
79  </script>
80  {% endblock %}
```

Verificar su funcionamiento

Método POST

En la vista de Contact

```
6  from django.http                    import JsonResponse
7  from django.views.decorators.http   import require_http_methods, require_POST
```

```
39  @require_POST
40  def get_cities(request):
41      selected_country = request.POST.get('country', '')
42      if selected_country == 'México':
43          cities = ['Ciudad de México', 'Guadalajara', 'Monterrey']
44      elif selected_country == 'USA':
45          cities = ['Nueva York', 'Los Ángeles', 'Chicago']
46      elif selected_country == 'Canada':
47          cities = ['Vancouver', 'Montreal', 'Quebec']
48      else:
49          cities = []
50      return JsonResponse({'cities':cities})
```

Crear el path en el archivo urls.py

```
7      path('get_countries/', views.get_countries, name='get_cou
8      path('get_cities/', views.get_cities, name='get_cities')
9  ], 'contact')
10
```

En el template contact.html

```
11          <select name="country" id="country" class='text-primary' ></se
12          <select name="city" id="city" class='text-primary' ></select>
13          <br>
```

Dentro del bloque `contentJs`

```
<script src='https://cdn.jsdelivr.net/npm/js-cookie@rc/dist/js.cookie.min.js'></script>
```

```
function csrfSafeMethod(method) {
        return (/^(GET|HEAD|OPTIONS|TRACE)$/.test(method));
    }
```

```
    $(function(){
        $('#country').change(function(){
            var selectedCountry = $(this).val();
            $.ajax({
                url: '/contact/get_cities/',
                type: 'POST',
                data: { 'country': selectedCountry },
                dataType: 'json',
                beforeSend: function(xhr, settings) {
                    if (!csrfSafeMethod(settings.type) && !this.crossDomain) {
                        xhr.setRequestHeader("X-CSRFToken", Cookies.get('csrftoken'));
                    }
                },
                success: function (data) {
                    var ciudadesSelect = $('#city');
                    ciudadesSelect.empty();
                    $.each(data.cities, function (index, ciudad) {
                        ciudadesSelect.append($('<option>').text(ciudad));
                    });
                }
            });
        });
    });
```

Verificar su funcionamiento

Práctica, utilizando JavaScript

(Opcional)

Función para obtener cualquier cookie almacenada

```
function getCookie(name) {
    let cookieArr = document.cookie.split(';');
    for(let i = 0; i < cookieArr.length; i++) {
        let cookiePair = cookieArr[i].split('=');
        if(name === cookiePair[0].trim()) {
            return decodeURIComponent(cookiePair[1]);
        }
    }
    return null;
}
```

Función para ejecutar un servicio con parámetros

```
function exec_service_rest(url, data) {
    return new Promise((resolve, reject) => {
        const csrftoken = getCookie('csrftoken');
        const xhr = new XMLHttpRequest();
        xhr.open('POST', url, true);
        xhr.setRequestHeader('Content-Type', 'application/json');
        xhr.setRequestHeader('X-CSRFToken', csrftoken);
        xhr.timeout = 5000;
        xhr.onload = function() {
            if (xhr.status === 200){
                try {
                    const resJSON = JSON.parse(xhr.responseText);
                    resolve(resJSON);
                }catch (error) {
                    reject(new Error('Error al parsar la respuesta JSON'));
                }
            }else {
                reject(new Error('Error en el ervicio: Status: ${xhr.status}'));
            }
        };
        xhr.onerror = function() {
            reject(new Error('Error en la red'));
        };
        xhr.ontimeout = function() {
            reject(new Error('Tiempo de espera agotado'));
        };
        xhr.send(JSON.stringify(data));
    });
}
```

Servicio get_countries

```
document.addEventListener('DOMContentLoaded', function(){
  const btnCountries = document.getElementById('btnCountries');
  btnCountries.addEventListener('click', function() {
    exec_service_rest('/contact/get_countries/', {})
    .then((data) => {
      const selectElement = document.getElementById('country');
      selectElement.innerHTML = '';
      for (let country of data.countries) {
        let newOption = document.createElement('option');
        newOption.value = country;
        newOption.text = country;
        selectElement.appendChild(newOption);
      }
    })
    .catch((error) =>{
      console.error(error);
    });
  });
```

Servicio get_cities

```
    const selectCountry = document.getElementById('country');
    selectCountry.addEventListener('change', function() {
      const country = selectCountry.value ;
      console.log(country);
      let parameters = {}
      parameters['country'] = country;
      exec_service_rest('/contact/get_cities/', parameters)
      .then((data) => {
        console.log(data);
        const selectElement = document.getElementById('city');
        selectElement.innerHTML = '';
        for (let city of data.cities) {
          let newOption = document.createElement('option');
          newOption.value = city;
          newOption.text = city;
          selectElement.appendChild(newOption);
        }
      })
      .catch((error) =>{
        console.error(error);
      });
    });
  });
```

Método de vista para servicio get_cities

```
@require_POST
def get_cities(request):
    data = json.loads(request.body)
    country = data.get('country', '')
    # country = request.POST.get('country', '')
    if country == 'México':
        cities = ['Ciudad de México', 'Guadalajara', 'Monterrey']
    elif country == 'USA':
        cities = ['Nueva York', 'Los Ángeles', 'Chicago']
    elif country == 'Canada':
        cities = ['Vancouver', 'Montreal', 'Quebec']
    else:
        cities = []
    return JsonResponse({'cities':cities})
```