Server Actions and Mutations

Server Actions

Server Actions, Next.js uygulamalarında sunucu tarafında veri işleme ve form gönderimi için kullanılan asenkron fonksiyonlardır.Uygulamanızın veri güncellemelerini ve form işlemlerini daha verimli bir şekilde yönetmenizi sağlar. Kapsamlı miras alma ve kademeli iyileştirme özellikleri ile kullanıcı deneyimini artırır. Sunucu eylemleri, kullanıcı etkileşimleri sonucu veri güncellemelerini gerçekleştirmek için hem Server hem de Client bileşenlerinden çağrılabilir.

Convention

Kullanım Şekli Bir Sunucu Eylemi, React'in "use server" yönergesiyle tanımlanabilir. Bu yönergeyi bir asenkron fonksiyonun en üstüne yerleştirerek o fonksiyonu Sunucu Eylemi olarak işaretleyebilir ya da ayrı bir dosyanın en üstüne yerleştirerek o dosyanın tüm dışa aktarımlarını Sunucu Eylemleri olarak işaretleyebilirsiniz.

Sunucu Bileşenleri, satır içi fonksiyon düzeyinde veya modül düzeyinde "use server" yönergesini kullanabilir. Bir Sunucu Eylemini satır içi tanımlamak için, fonksiyonun gövdesinin en üstüne "use server" ekleyin.

app/page.js

```
1  export default function Page() {
2    // Server Action
3    async function create() {
4      'use server'
5      // Mutate data
6    }
7
8    return '...'
9  }
```

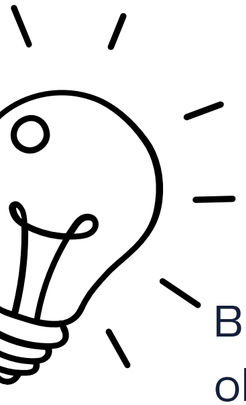
İstemci Bileşenleri Bir İstemci Bileşeninde Sunucu Eylemi çağırmak için, yeni bir dosya oluşturun ve dosyanın en üstüne "use server" yönergesini ekleyin. Dosya içindeki tüm dışa aktarılan fonksiyonlar, hem İstemci hem de Sunucu Bileşenlerinde yeniden kullanılabilen Sunucu Eylemleri olarak işaretlenecektir.

app/actions.js

```
1  'use server'
2
3  export async function create() {}
```

app/ui/button.js

```
1  'use client'
2
3  import { create } from '@/app/actions'
4
5  export function Button() {
6    return <button onClick={() => create()}>Create</button>
7  }
```



Passing actions as props

Bir Sunucu Eylemini, bir İstemci Bileşenine prop olarak da geçirebilirsiniz:

```
<ClientComponent updateItemAction={updateItem} />
```

app/client-component.js — JavaScript

```
'use client'

export default function ClientComponent({ updateItemAction }) {
  return <form action={updateItemAction}>{/* ... */}</form>
}
```

Genellikle, Next.js'de, client-component.js dosyasındaki updateItemAction, istemci-sunucu sınırlarında serileştirilemeyen bir fonksiyon olduğu için dikkat çekebilir. Ancak, adı action olan veya Action ile biten prop'ların Sunucu Eylemi aldığı varsayılır. Bu, sadece bir varsayımdır; çünkü Next.js, gerçekte bu prop'un bir Sunucu Eylemi mi yoksa sıradan bir fonksiyon mu olduğunu belirlemez. Yine de uygulama çalıştığında yapılan tür kontrolü, istemci bileşenine yanlışlıkla bir fonksiyon geçirip geçirmediğinizi kontrol eder.

Behavior

- **Form Gönderimi:** Sunucu eylemleri, bir <form> öğesindeki action niteliği ile çağrılabilir.

- **Kademeli İyileştirme:** Sunucu Bileşenleri, varsayılan olarak kademeli iyileştirmeyi destekler. Bu, JavaScript yüklenmemiş veya devre dışı bırakılmış olsa bile formun gönderilebileceği anlamına gelir.

- **İstemci Bileşenleri:** İstemci Bileşenlerinde, Sunucu Eylemlerini çağıran formlar, JavaScript henüz yüklenmemişse gönderimleri sıraya alır ve istemci hidratasyonunu önceliklendirir. Hidratasyondan sonra, form gönderiminde tarayıcı yenilenmez.

- **Diğer Çağrılar:** Sunucu Eylemleri yalnızca <form> ile sınırlı değildir. Olay işleyicileri, useEffect, üçüncü parti kütüphaneler ve <button> gibi diğer form öğelerinden de çağrılabilir.

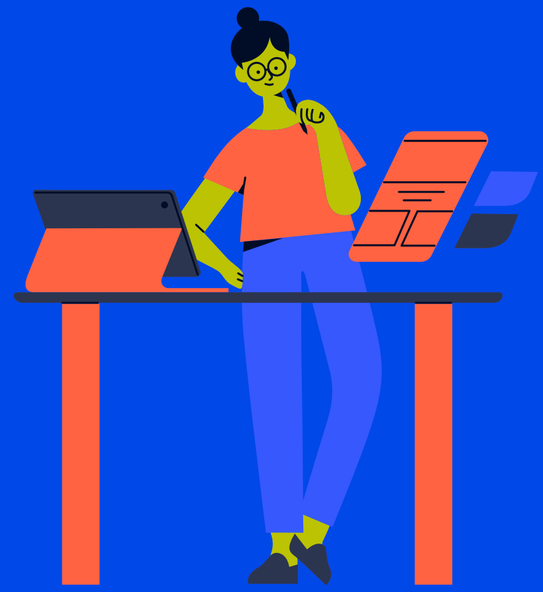
- **Önbellekleme ve Yeniden Doğrulama:** Sunucu Eylemleri, Next.js'in önbellekleme ve yeniden doğrulama mimarisi ile entegre olur. Bir eylem çağrıldığında, Next.js güncellenmiş UI ve yeni veriyi tek bir sunucu gidiş dönüşünde döndürebilir.

- **HTTP Yöntemi:** Sunucu Eylemleri arka planda POST yöntemini kullanır ve yalnızca bu yöntemle çağrılabilirler.

- **Serileştirilebilirlik:** Sunucu Eylemleri'nin argümanları ve dönüş değerleri React tarafından serileştirilebilir olmalıdır. Serileştirilebilir argümanlar ve değerler için React belgelerine başvurulmalıdır.

- **Fonksiyonlar:** Sunucu Eylemleri fonksiyondur, bu da uygulamanızın her yerinde yeniden kullanılabilecekleri anlamına gelir.

- **Miras Alma:** Sunucu Eylemleri, kullanıldıkları sayfanın veya düzenin çalışma zamanını ve Rota Segment Yapılandırmasını (örneğin maxDuration gibi alanlar) miras alır.



Form Gönderimi:

UserProfile bileşeninde bir form tanımlanmış. Form gönderildiğinde, updateUserWithId fonksiyonu çağrılır.
Bu fonksiyon, updateUser fonksiyonunun userId argümanıyla bağlanmış bir versiyonudur. bind metodu, userId'yi otomatik olarak updateUser fonksiyonuna geçirecek şekilde ayarlıyor.

Form gönderildiğinde, updateUser fonksiyonu çağrıldığında iki argüman alır: userId ve formdan gelen FormData nesnesi. formData nesnesi, form içindeki tüm alanların verilerini içerir. Örneğin, name alanındaki değer formData.get('name') ile alınabilir.UpdateUser fonksiyonu, kullanıcının adı güncellenirken kullanılmak üzere her iki argümanı da işleyebilir.

app/client-component.js — JavaScript

```
1  'use client'
2
3  import { updateUser } from './actions'
4
5  export function UserProfile({ userId }) {
6    const updateUserWithId = updateUser.bind(null, userId)
7
8    return (
9      <form action={updateUserWithId}>
10       <input type="text" name="name" />
11       <button type="submit">Update User Name</button>
12     </form>
13   )
14 }
```

app/actions.js

```
1  'use server'
2
3  export async function updateUser(userId, formData) {}
```

Thank You