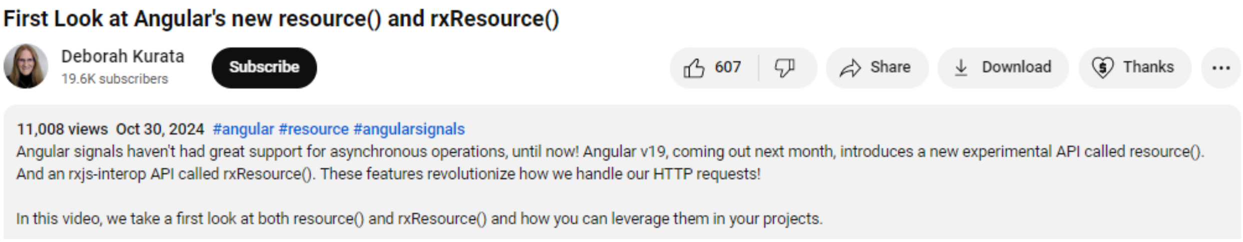
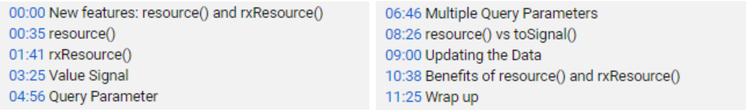


# First Look at Angular's new resource() and rxResource()

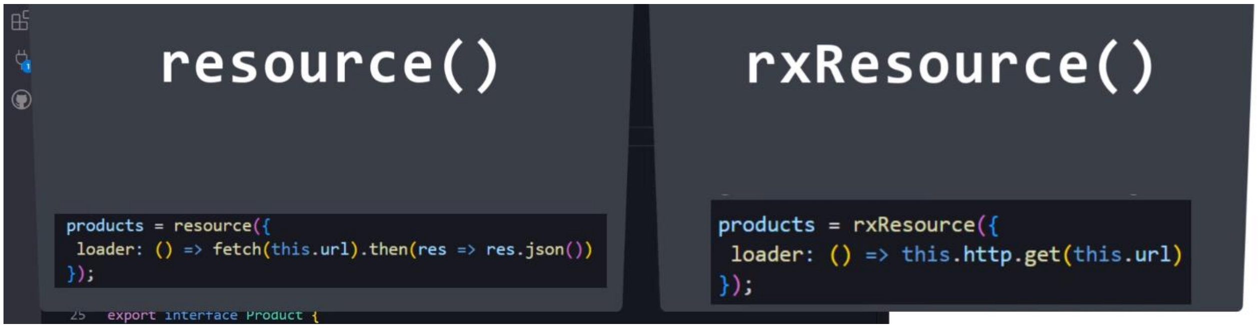
11,008 views Oct 30, 2024 #angular #resource #angularsignals

Angular signals haven't had great support for asynchronous operations, until now! Angular v19, coming out next month, introduces a new experimental API called resource(). And an rxjs-interop API called rxResource(). These features revolutionize how we handle our HTTP requests!

In this video, we take a first look at both resource() and rxResource() and how you can leverage them in your projects.



00:00 New features: resource() and rxResource()
00:35 resource()
01:41 rxResource()
03:25 Value Signal
04:56 Query Parameter
06:46 Multiple Query Parameters
08:26 resource() vs toSignal()
09:00 Updating the Data
10:38 Benefits of resource() and rxResource()
11:25 Wrap up



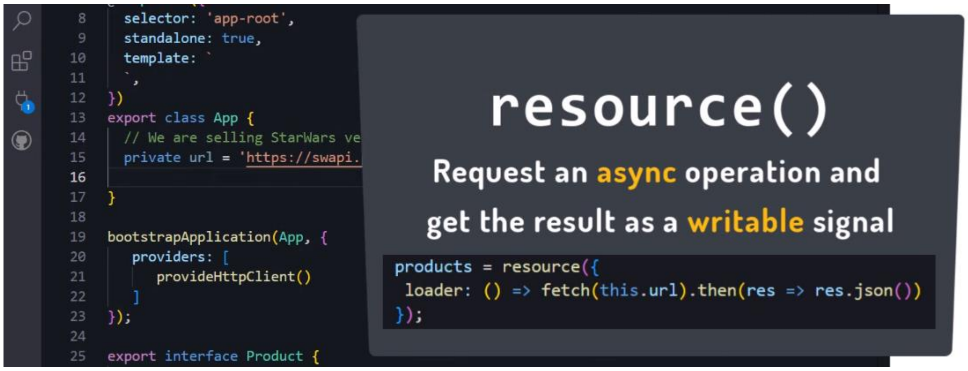
resource()

```
products = resource({
  loader: () => fetch(this.url).then(res => res.json())
});
```

rxResource()

```
products = rxResource({
  loader: () => this.http.get(this.url)
});
```

```
export interface Product {
```

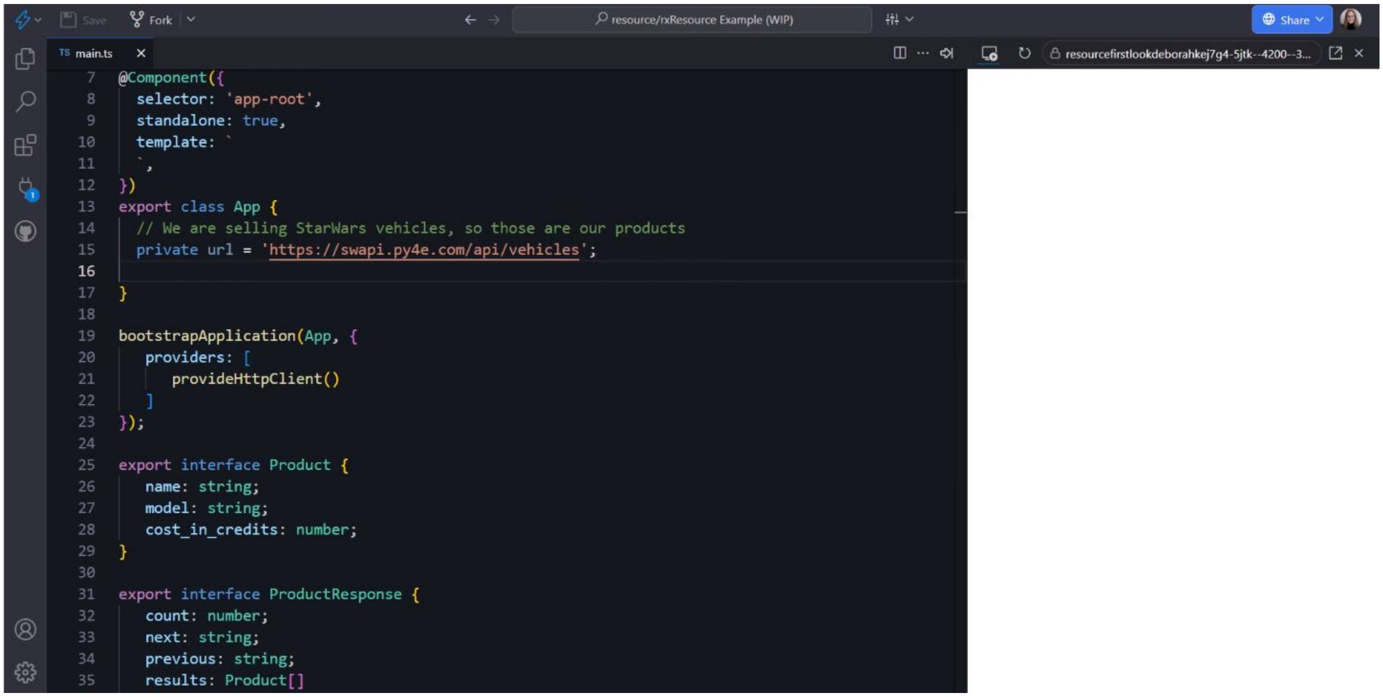


```
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars ve
  private url = 'https://swapi.

}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
   ]
});

export interface Product {
```

resource()

Request an async operation and get the result as a writable signal

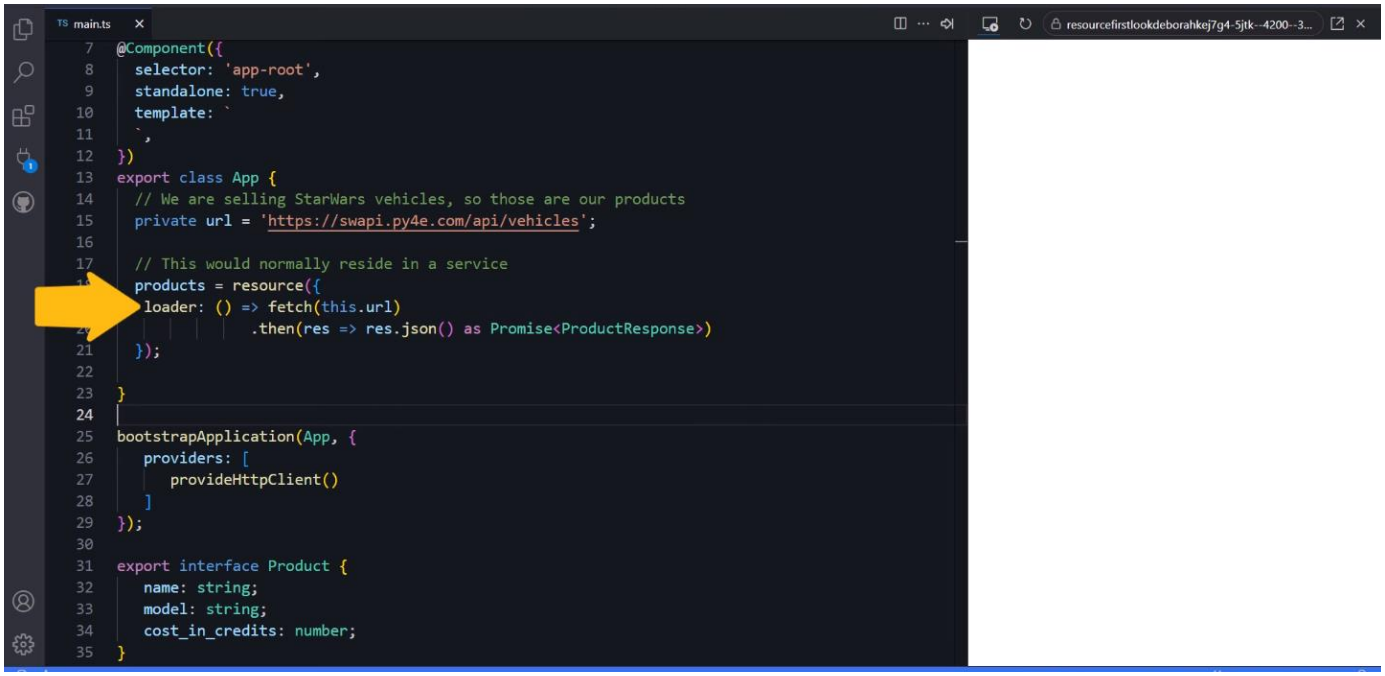
```
products = resource({
  loader: () => fetch(this.url).then(res => res.json())
});
```



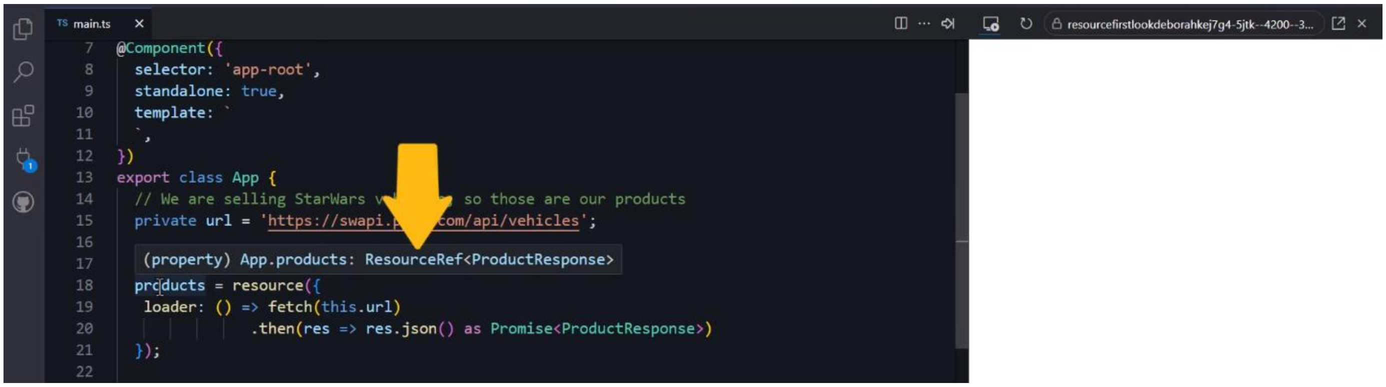
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g4-5jtk--4200--3...

```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';

}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
   ]
});

export interface Product {
   name: string;
   model: string;
   cost_in_credits: number;
}

export interface ProductResponse {
   count: number;
   next: string;
   previous: string;
   results: Product[]
```

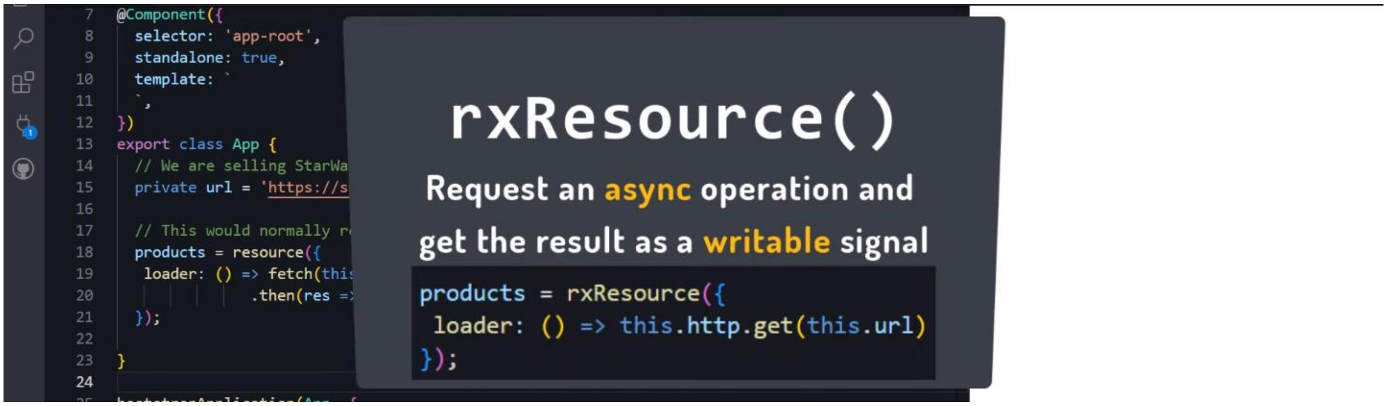
We want to retrieve products when our component is initialized,



```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';

  // This would normally reside in a service
  products = resource({
   loader: () => fetch(this.url)
                .then(res => res.json() as Promise<ProductResponse>)
  });

}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
   ]
});

export interface Product {
   name: string;
   model: string;
   cost_in_credits: number;
}
```



```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars v          so those are our products
  private url = 'https://swapi.p       com/api/vehicles';

   (property) App.products: ResourceRef<ProductResponse>
  products = resource({
   loader: () => fetch(this.url)
                .then(res => res.json() as Promise<ProductResponse>)
  });
```



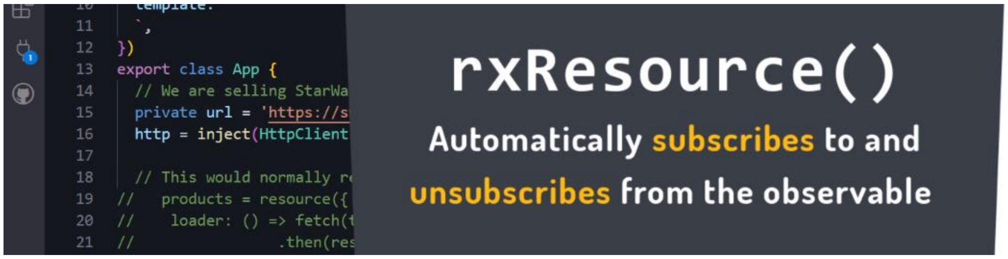
```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWa
  private url = 'https://s

  // This would normally r
  products = resource({
   loader: () => fetch(thi
                .then(res =
  });

}
```

rxResource()

Request an async operation and get the result as a writable signal

```
products = rxResource({
 loader: () => this.http.get(this.url)
});
```

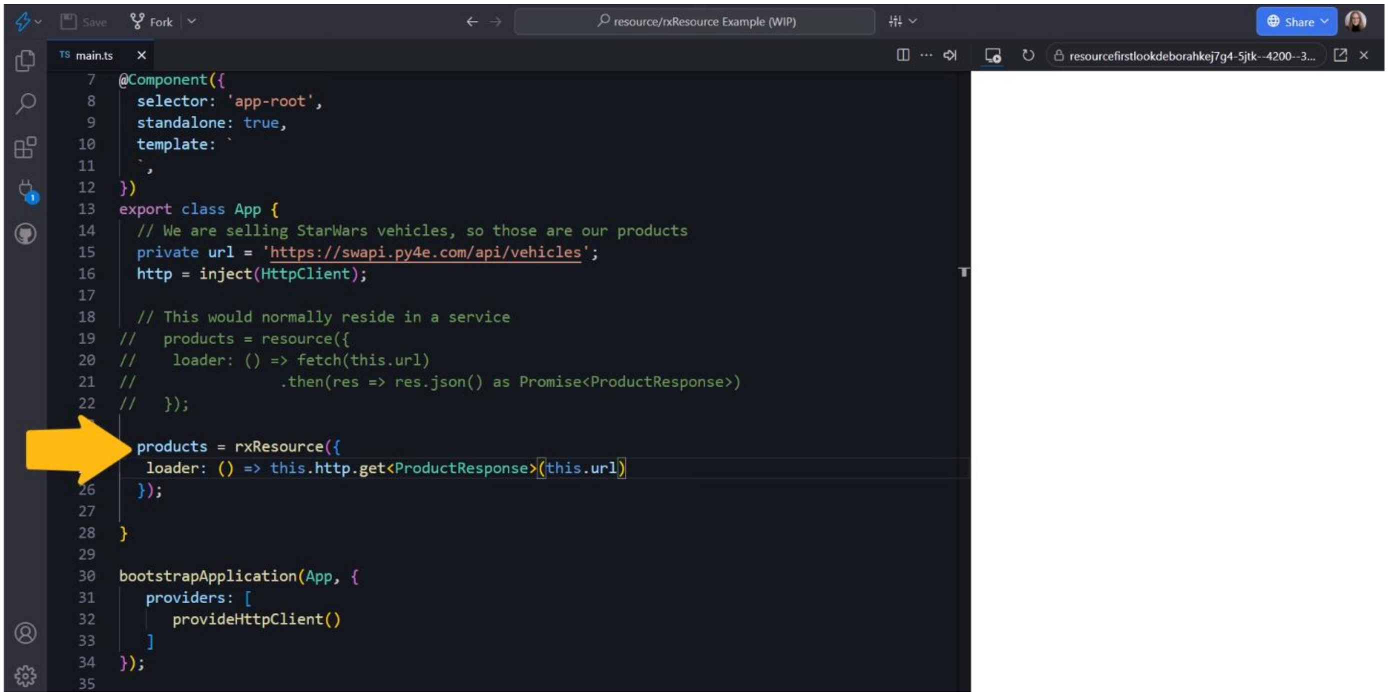
You can use **rxResource()** if you want to still use the HTTP client to return an Observable instead of a Promise.



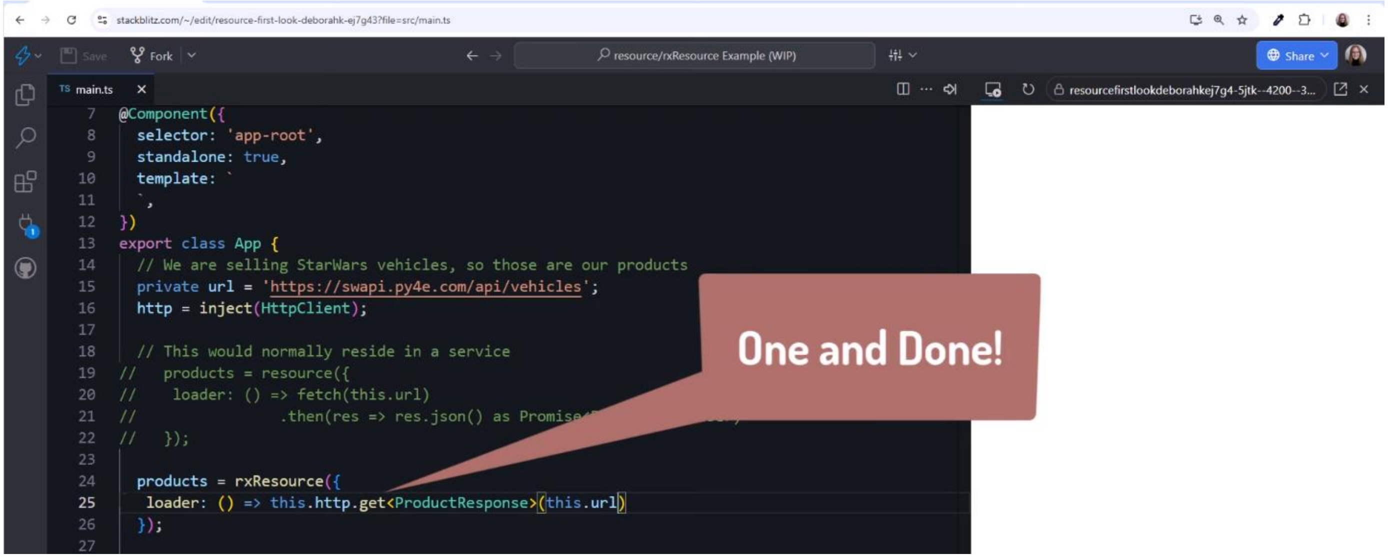
```
  `,
})
export class App {
  // We are selling StarWa
  private url = 'https://s
  http = inject(HttpClient

  // This would normally r
//   products = resource({
//     loader: () => fetch(t
//                  .then(res
```

rxResource()

Automatically subscribes to and unsubscribes from the observable


Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g4-5jtk--4200--3...

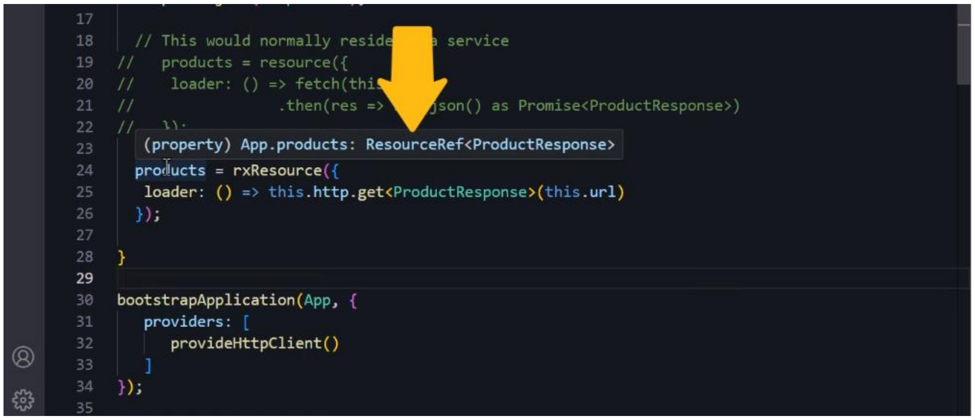
```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
    loader: () => this.http.get<ProductResponse>(this.url)
  });

}

bootstrapApplication(App, {
  providers: [
    provideHttpClient()
  ]
});
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g4-5jtk--4200--3...

```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                .then(res => res.json() as Promise
//   });

  products = rxResource({
    loader: () => this.http.get<ProductResponse>(this.url)
  });
```

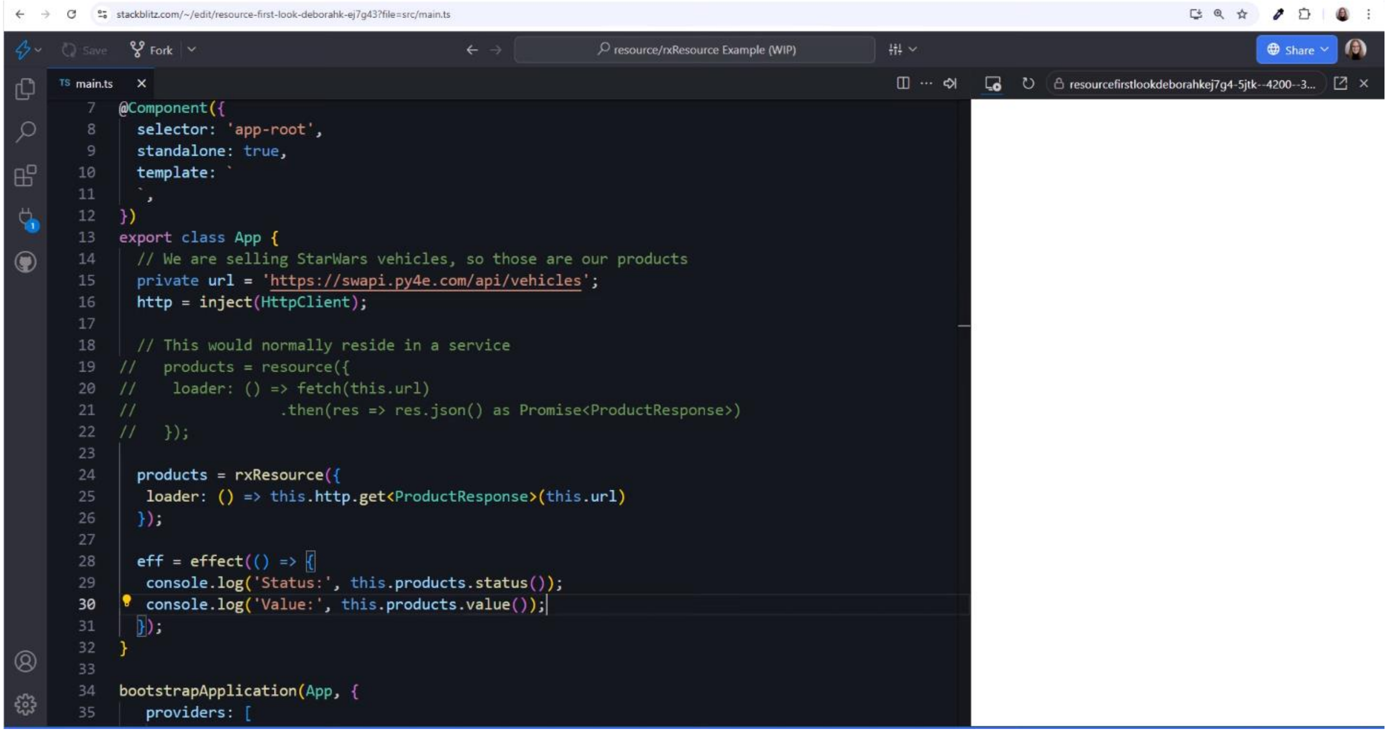
One and Done!

This only emits once and completes.



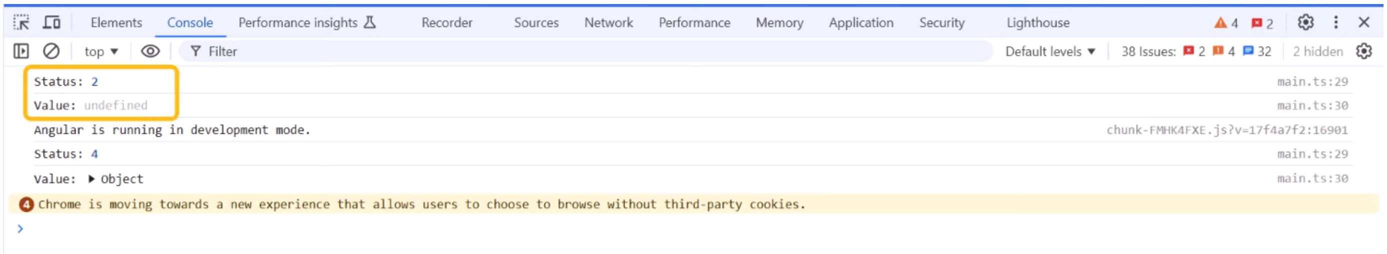
```
  // This would normally reside   a service
//   products = resource({
//     loader: () => fetch(thi
//                .then(res =>    json() as Promise<ProductResponse>)
//   });
  (property) App.products: ResourceRef<ProductResponse>
  products = rxResource({
    loader: () => this.http.get<ProductResponse>(this.url)
  });

}

bootstrapApplication(App, {
  providers: [
    provideHttpClient()
  ]
});
```

We now need to get the value of the HttpResponse as a signal as below

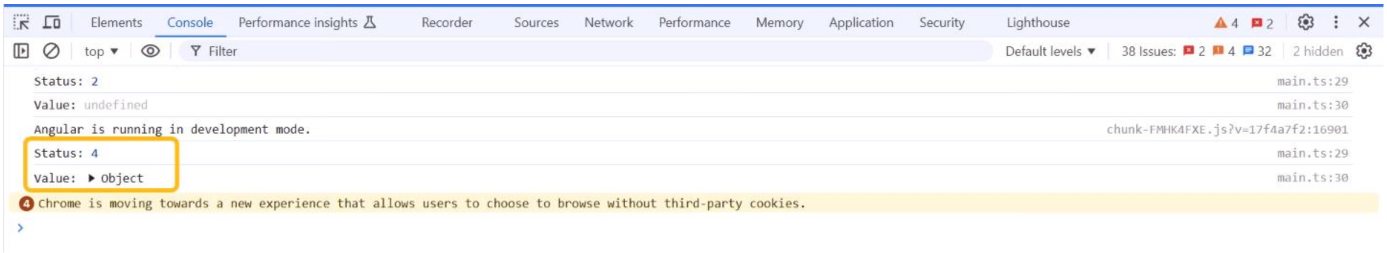


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g4-5jtk--4200--3...

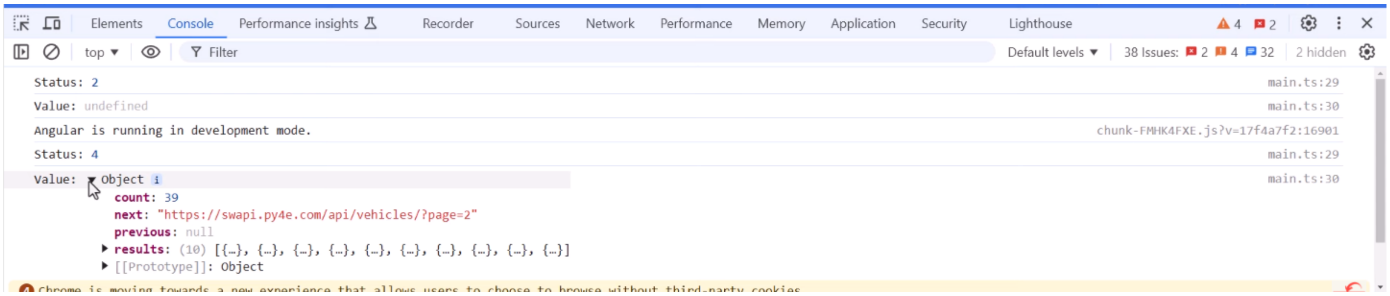
```typescript
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  eff = effect(() => {
    console.log('Status:', this.products.status());
    console.log('Value:', this.products.value());
  });
}

bootstrapApplication(App, {
  providers: [
```



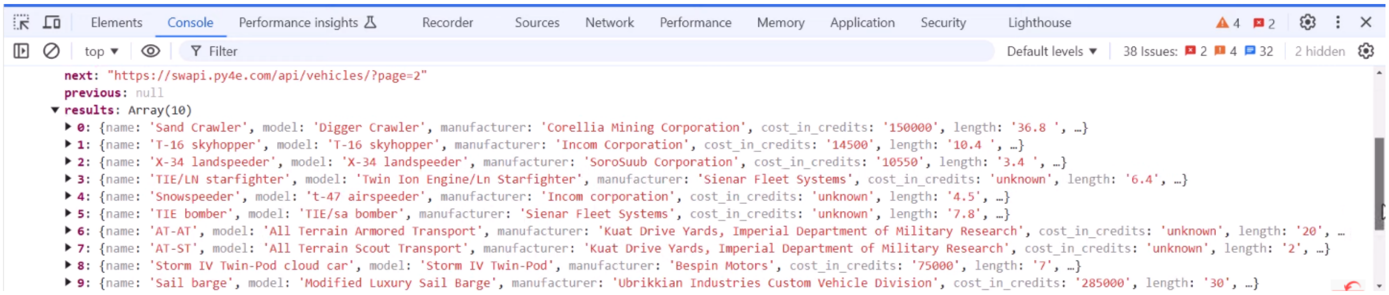
```
Status: 2                                                          main.ts:29
Value: undefined                                                   main.ts:30
Angular is running in development mode.       chunk-FMHK4FXE.js?v=17f4a7f2:16901
Status: 4                                                          main.ts:29
Value: ▶ Object                                                    main.ts:30
4 Chrome is moving towards a new experience that allows users to choose to browse without third-party cookies.
>
```



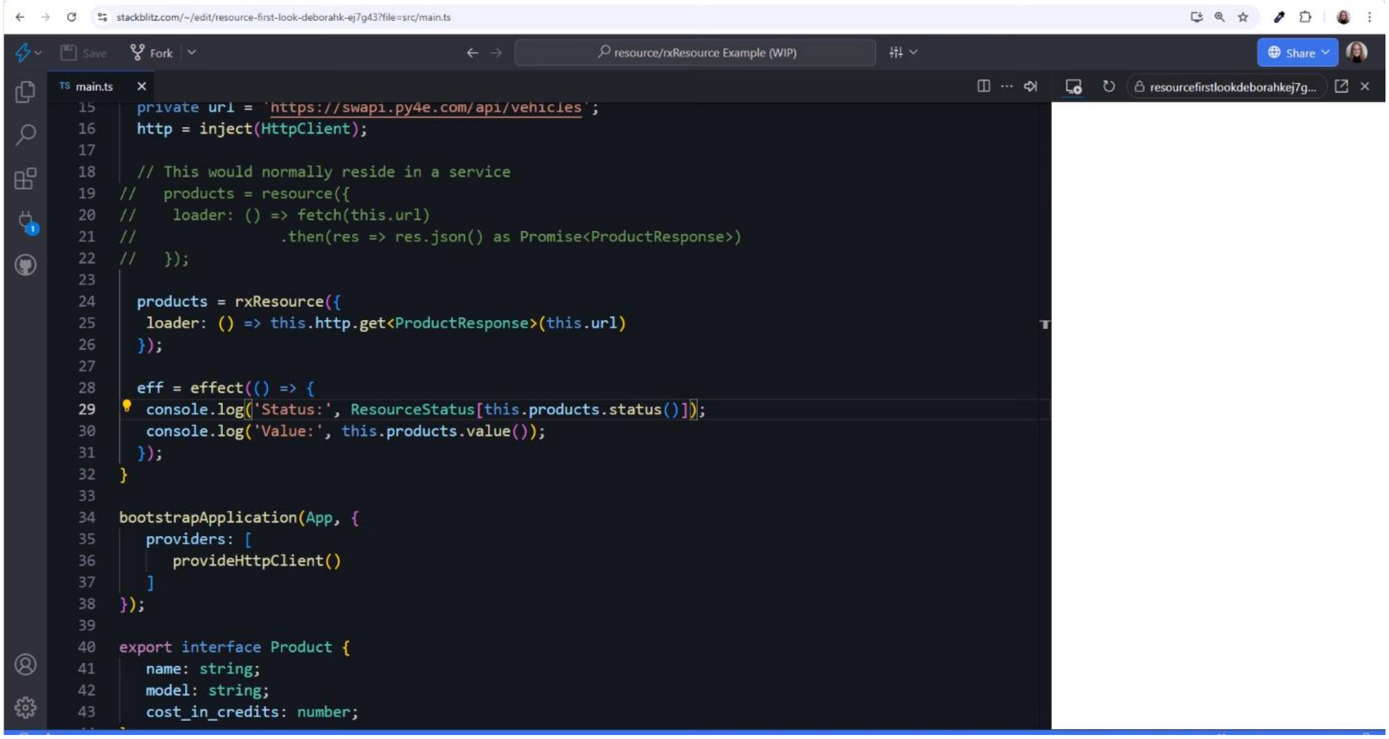
```
Status: 2                                                          main.ts:29
Value: undefined                                                   main.ts:30
Angular is running in development mode.       chunk-FMHK4FXE.js?v=17f4a7f2:16901
Status: 4                                                          main.ts:29
Value: ▶ Object                                                    main.ts:30
4 Chrome is moving towards a new experience that allows users to choose to browse without third-party cookies.
>
```



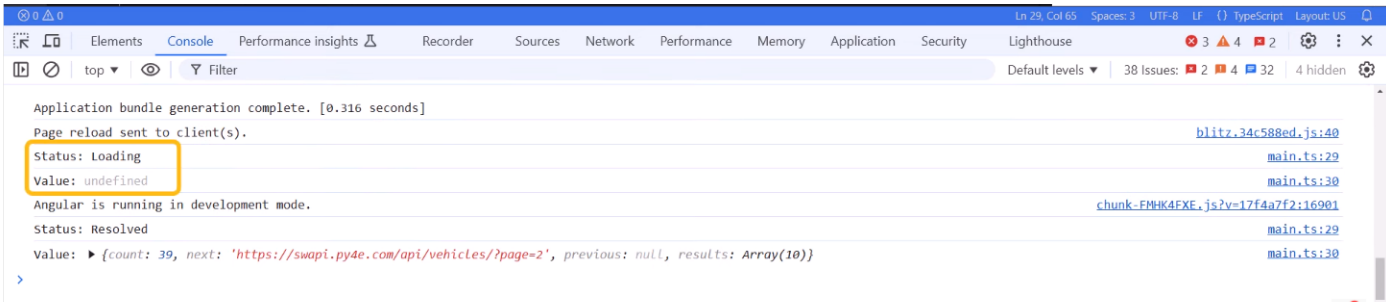
```
Status: 2                                                          main.ts:29
Value: undefined                                                   main.ts:30
Angular is running in development mode.       chunk-FMHK4FXE.js?v=17f4a7f2:16901
Status: 4                                                          main.ts:29
Value: ▼ Object i                                                  main.ts:30
          count: 39
          next: "https://swapi.py4e.com/api/vehicles/?page=2"
          previous: null
        ▶ results: (10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
        ▶ [[Prototype]]: Object
4 Chrome is moving towards a new experience that allows users to choose to browse without third-party cookies.
```



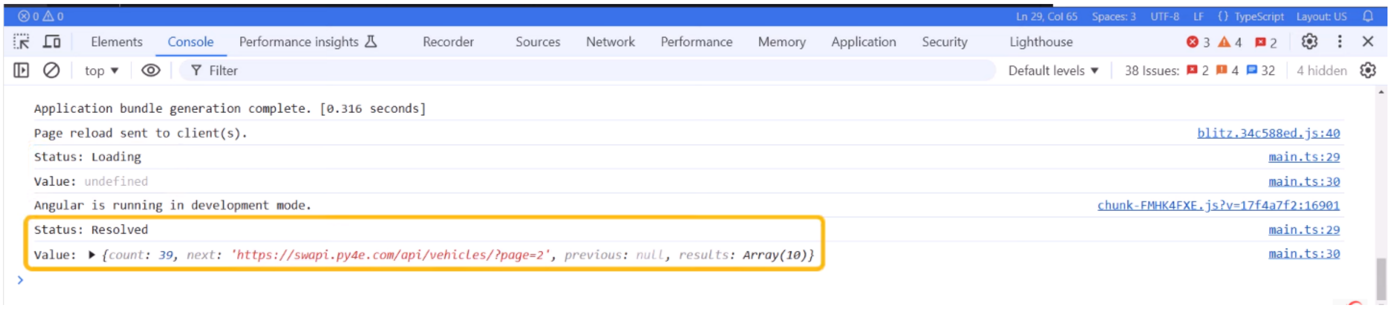
```
      next: "https://swapi.py4e.com/api/vehicles/?page=2"
      previous: null
    ▼ results: Array(10)
      ▶ 0: {name: 'Sand Crawler', model: 'Digger Crawler', manufacturer: 'Corellia Mining Corporation', cost_in_credits: '150000', length: '36.8 ', …}
      ▶ 1: {name: 'T-16 skyhopper', model: 'T-16 skyhopper', manufacturer: 'Incom Corporation', cost_in_credits: '14500', length: '10.4 ', …}
      ▶ 2: {name: 'X-34 landspeeder', model: 'X-34 landspeeder', manufacturer: 'SoroSuub Corporation', cost_in_credits: '10550', length: '3.4 ', …}
      ▶ 3: {name: 'TIE/LN starfighter', model: 'Twin Ion Engine/Ln Starfighter', manufacturer: 'Sienar Fleet Systems', cost_in_credits: 'unknown', length: '6.4', …}
      ▶ 4: {name: 'Snowspeeder', model: 't-47 airspeeder', manufacturer: 'Incom corporation', cost_in_credits: 'unknown', length: '4.5', …}
      ▶ 5: {name: 'TIE bomber', model: 'TIE/sa bomber', manufacturer: 'Sienar Fleet Systems', cost_in_credits: 'unknown', length: '7.8', …}
      ▶ 6: {name: 'AT-AT', model: 'All Terrain Armored Transport', manufacturer: 'Kuat Drive Yards, Imperial Department of Military Research', cost_in_credits: 'unknown', length: '20', …
      ▶ 7: {name: 'AT-ST', model: 'All Terrain Scout Transport', manufacturer: 'Kuat Drive Yards, Imperial Department of Military Research', cost_in_credits: 'unknown', length: '2', …}
      ▶ 8: {name: 'Storm IV Twin-Pod cloud car', model: 'Storm IV Twin-Pod', manufacturer: 'Bespin Motors', cost_in_credits: '75000', length: '7', …}
      ▶ 9: {name: 'Sail barge', model: 'Modified Luxury Sail Barge', manufacturer: 'Ubrikkian Industries Custom Vehicle Division', cost_in_credits: '285000', length: '30', …}
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

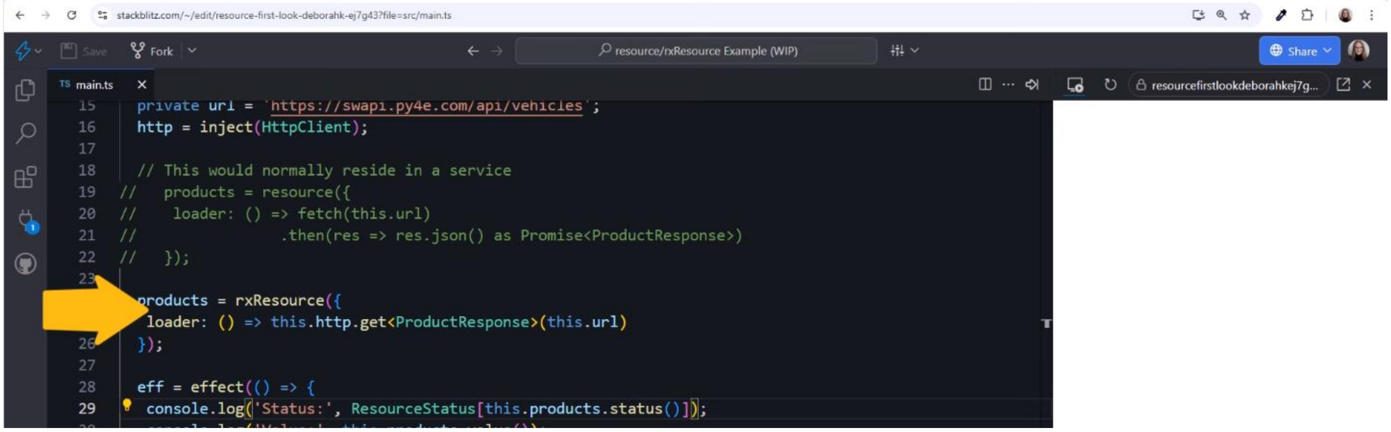
```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });
}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
   ]
});

export interface Product {
   name: string;
   model: string;
   cost_in_credits: number;
```



```
Application bundle generation complete. [0.316 seconds]
Page reload sent to client(s).                                             blitz.34c588ed.js:40
Status: Loading                                                                  main.ts:29
Value: undefined                                                                 main.ts:30
Angular is running in development mode.                      chunk-FMHK4FXE.js?v=17f4a7f2:16901
Status: Resolved                                                                 main.ts:29
Value: ▶ {count: 39, next: 'https://swapi.py4e.com/api/vehicles/?page=2', previous: null, results: Array(10)}   main.ts:30
```

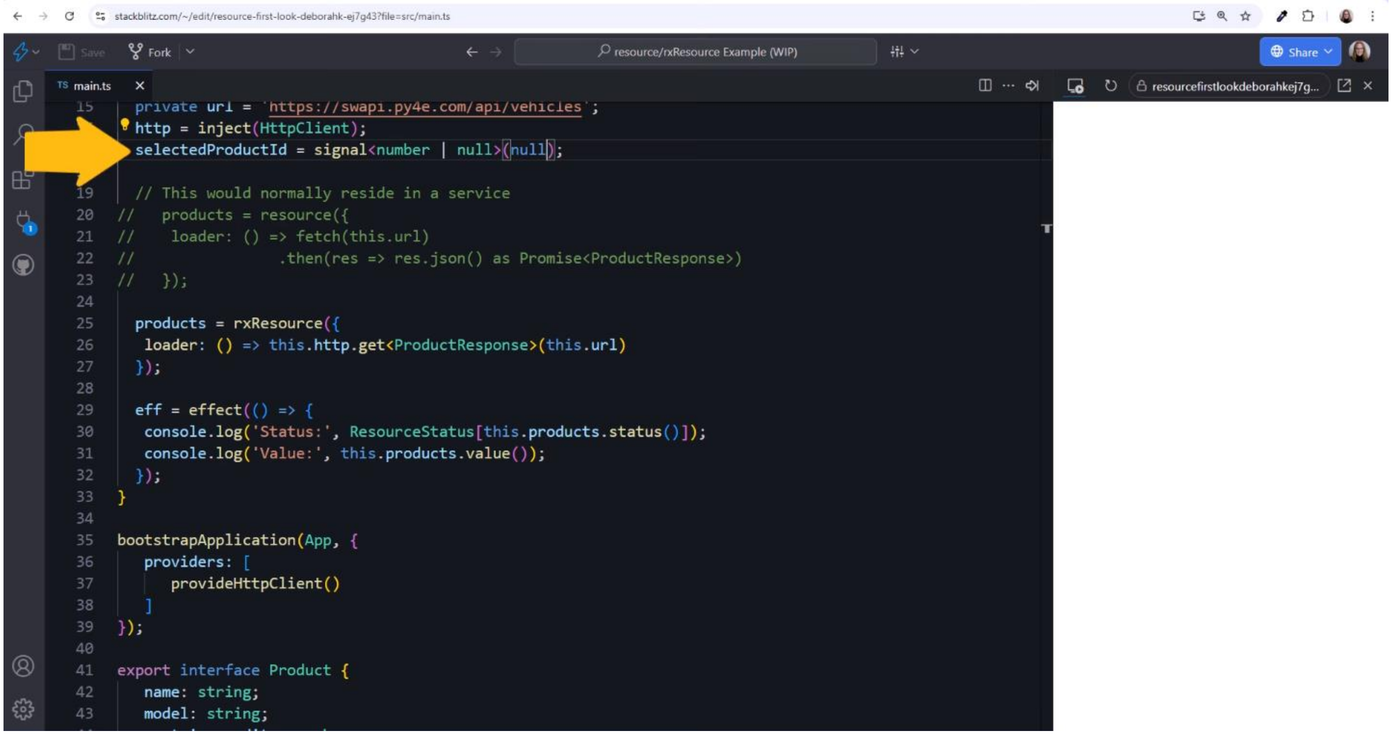


```
Application bundle generation complete. [0.316 seconds]
Page reload sent to client(s).                                             blitz.34c588ed.js:40
Status: Loading                                                                  main.ts:29
Value: undefined                                                                 main.ts:30
Angular is running in development mode.                      chunk-FMHK4FXE.js?v=17f4a7f2:16901
Status: Resolved                                                                 main.ts:29
Value: ▶ {count: 39, next: 'https://swapi.py4e.com/api/vehicles/?page=2', previous: null, results: Array(10)}   main.ts:30
```

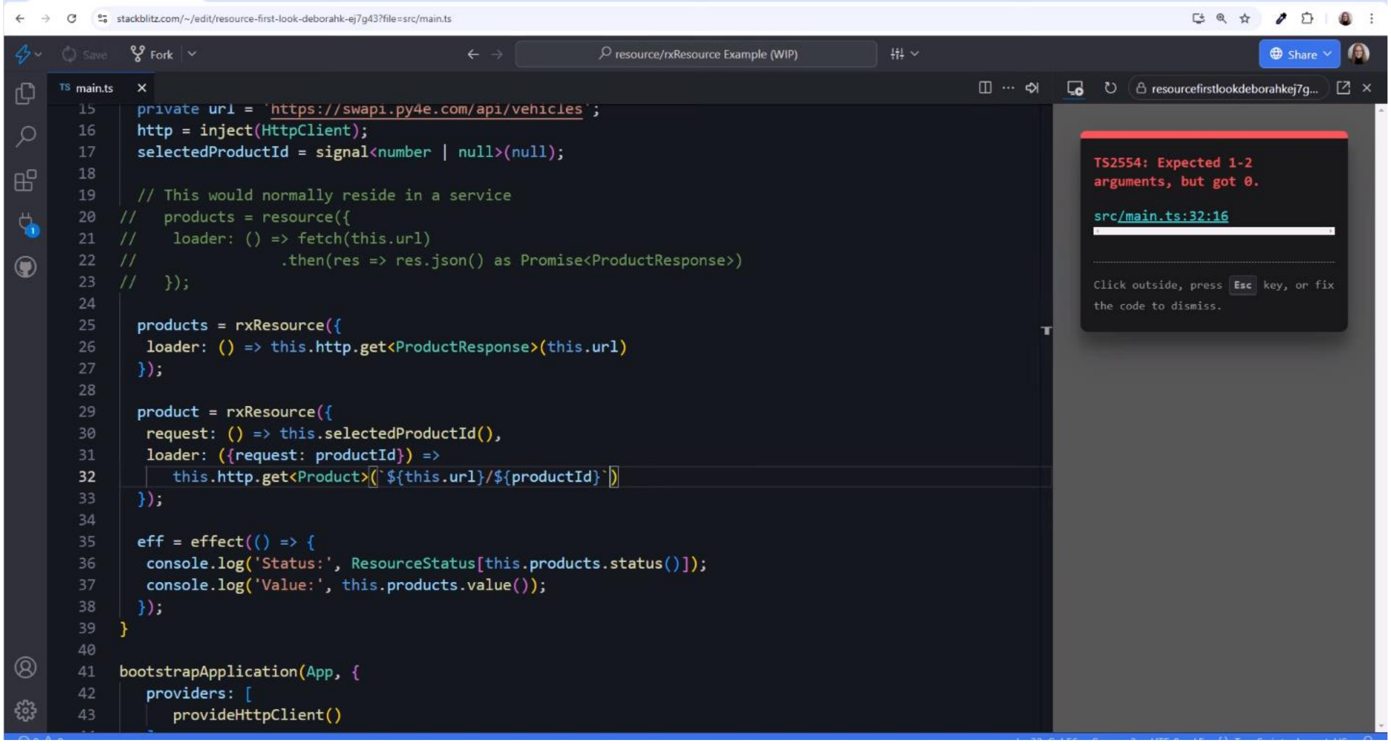

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
```

Next, let us see how we can pass in parameters for our API call as below

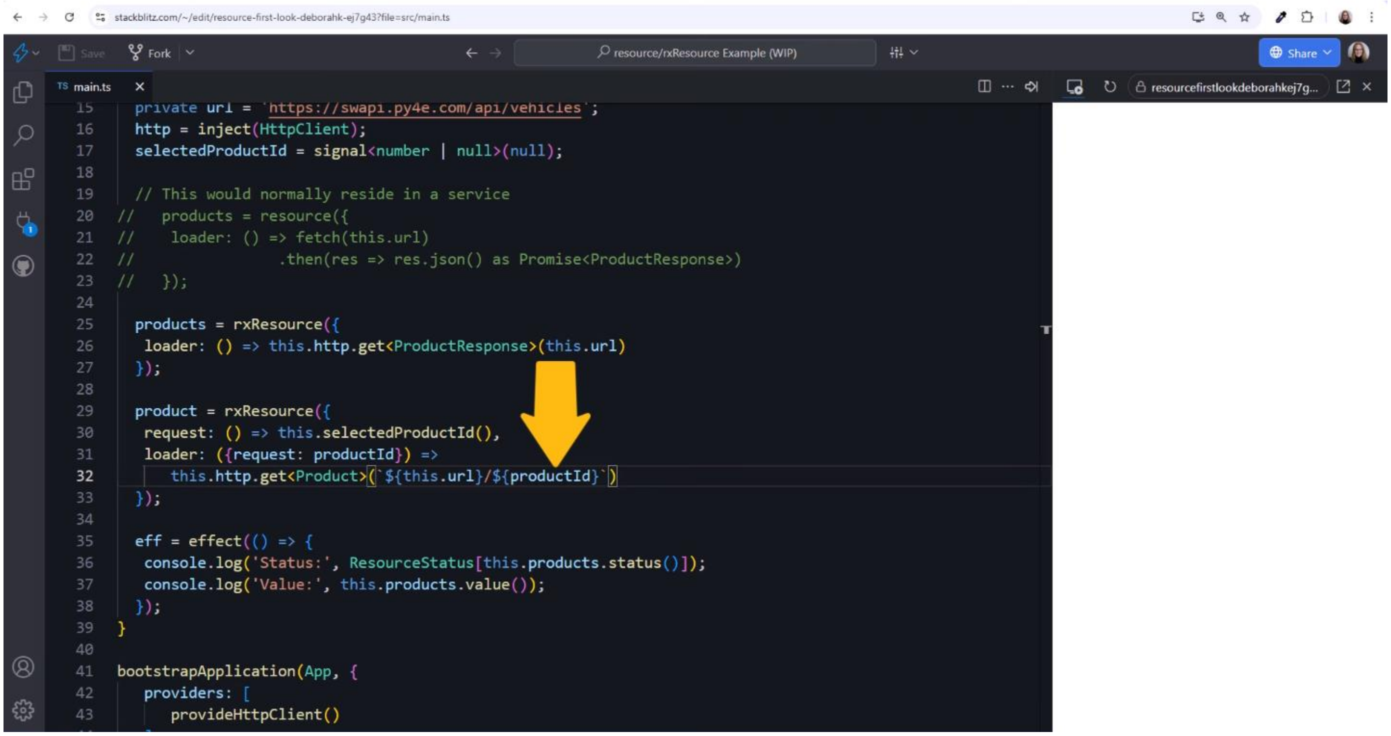

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });
}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
   ]
});

export interface Product {
   name: string;
   model: string;
```

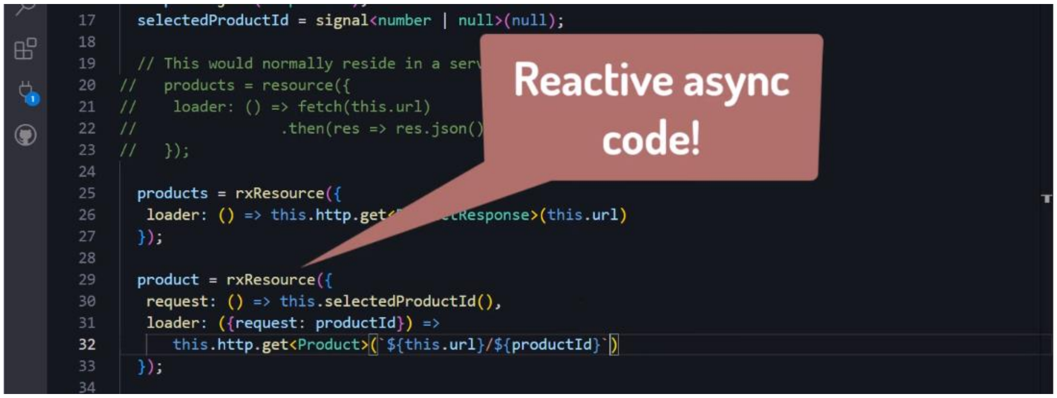

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => this.selectedProductId(),
   loader: ({request: productId}) =>
      this.http.get<Product>(`${this.url}/${productId}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });
}

bootstrapApplication(App, {
   providers: [
      provideHttpClient()
```

```
TS2554: Expected 1-2
arguments, but got 0.

src/main.ts:32:16

Click outside, press Esc key, or fix
the code to dismiss.
```

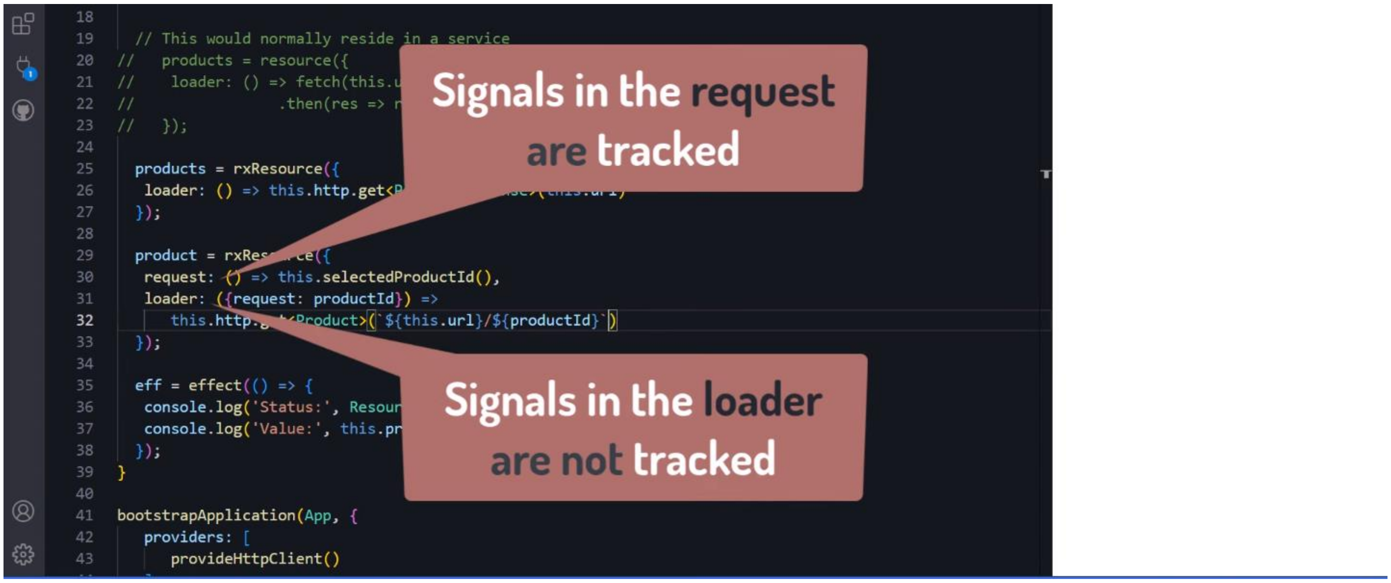

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);

  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
    loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
    request: () => this.selectedProductId(),
    loader: ({request: productId}) =>
      this.http.get<Product>(`${this.url}/${productId}`)
  });

  eff = effect(() => {
    console.log('Status:', ResourceStatus[this.products.status()]);
    console.log('Value:', this.products.value());
  });
}

bootstrapApplication(App, {
  providers: [
    provideHttpClient()
```



Reactive async code!

```
  selectedProductId = signal<number | null>(null);

  // This would normally reside in a serv
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json()
//   });

  products = rxResource({
    loader: () => this.http.get<     ctResponse>(this.url)
  });

  product = rxResource({
    request: () => this.selectedProductId(),
    loader: ({request: productId}) =>
      this.http.get<Product>(`${this.url}/${productId}`)
  });
```

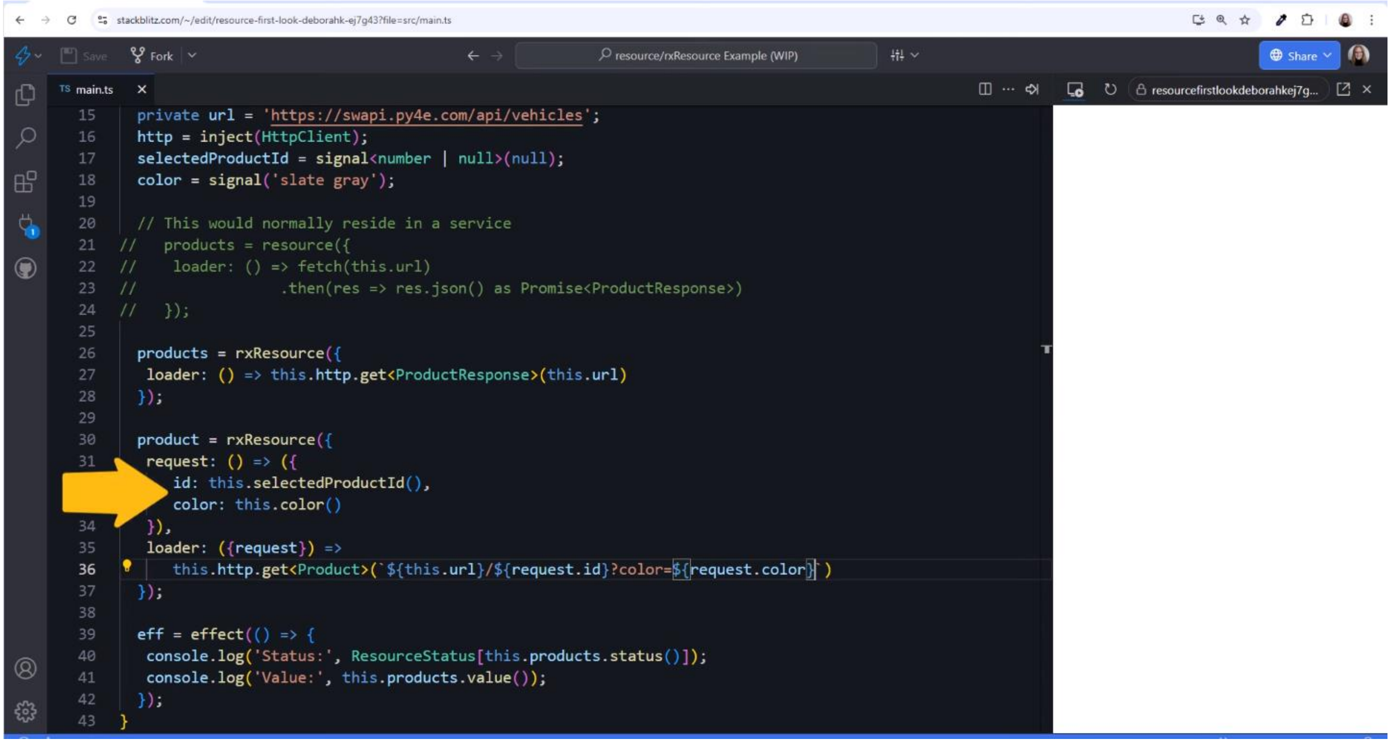


Signals in the request are tracked

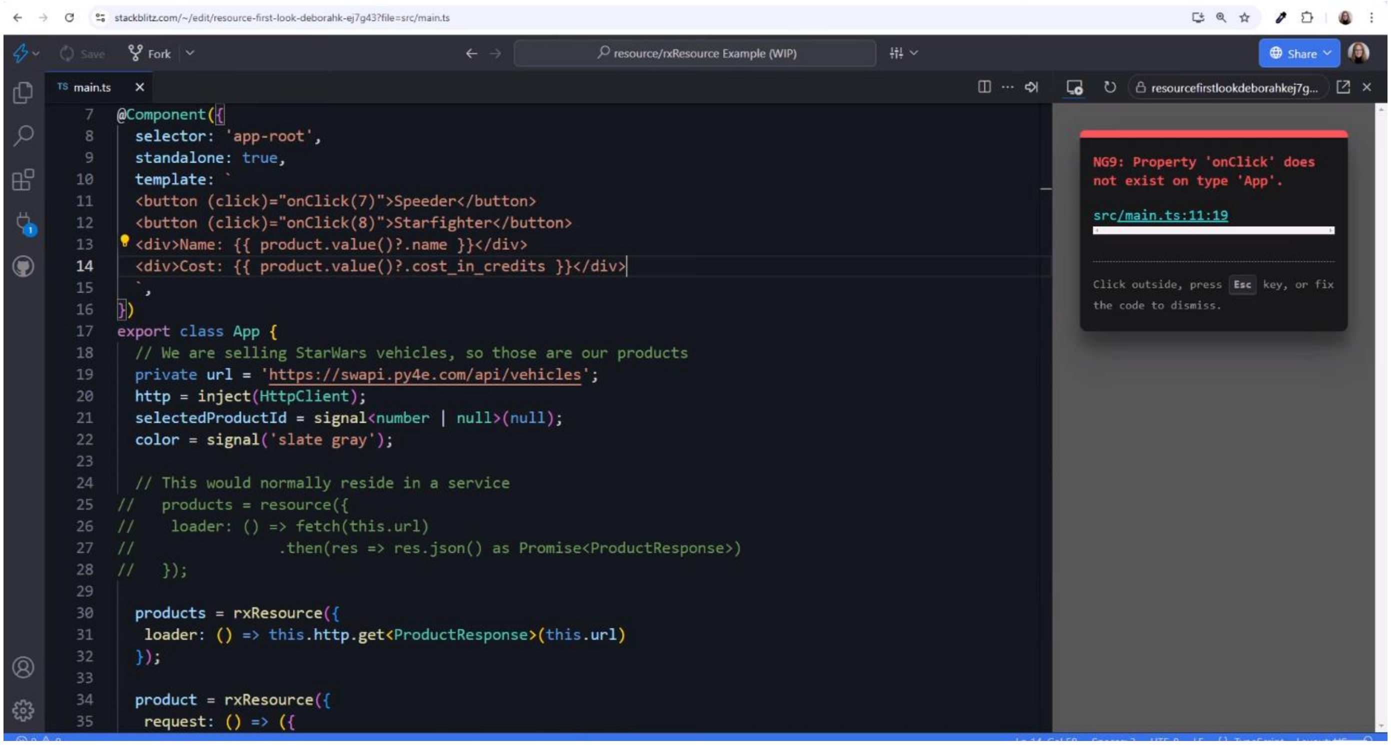
Signals in the loader are not tracked

```
  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.u
//                 .then(res => r
//   });

  products = rxResource({
    loader: () => this.http.get<P
  });

  product = rxRes     ce({
    request: () => this.selectedProductId(),
    loader: ({request: productId}) =>
      this.http.   <Product>(`${this.url}/${productId}`)
  });

  eff = effect(() => {
    console.log('Status:', Resour
    console.log('Value:', this.pr
  });
}

bootstrapApplication(App, {
  providers: [
    provideHttpClient()
```

We can also pass in multiple query parameters as below


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);
  color = signal('slate gray');

  // This would normally reside in a service
//   products = resource({
//    loader: () => fetch(this.url)
//                .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });
}
```

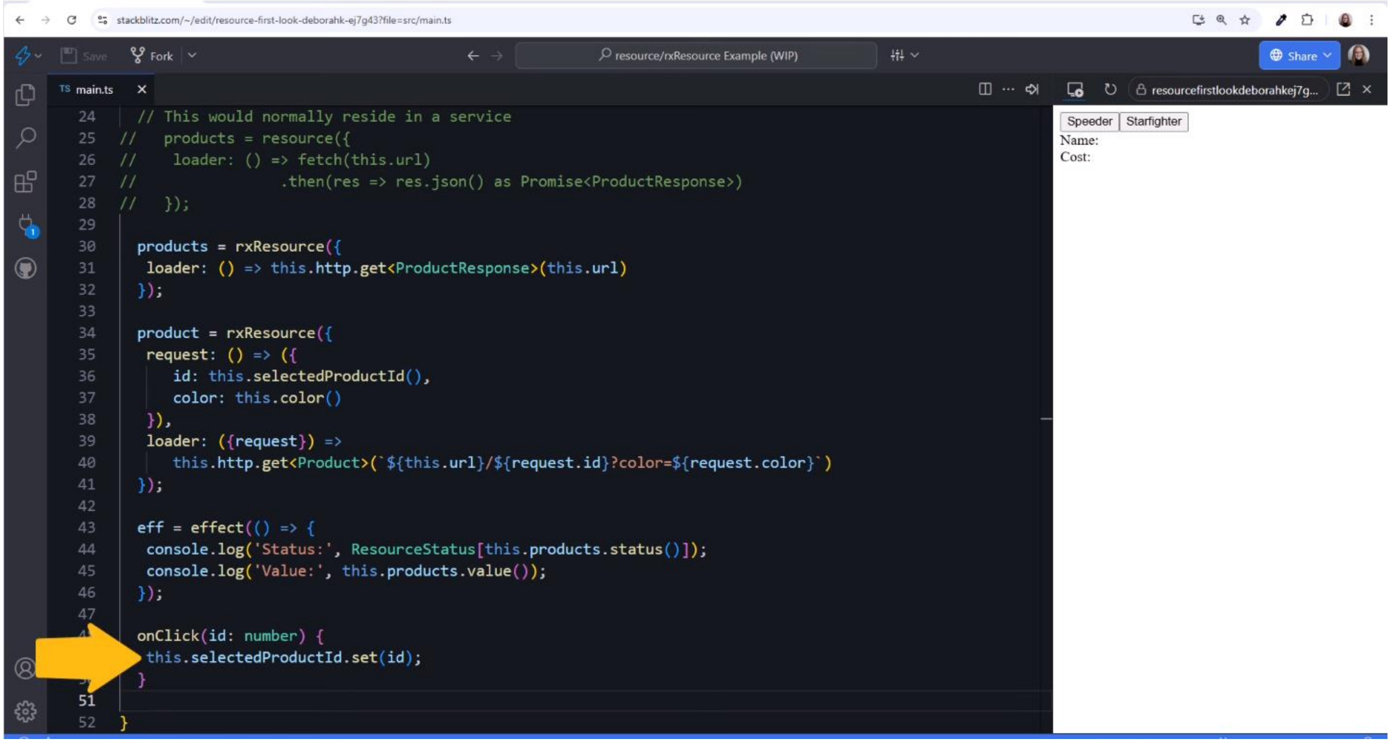

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
@Component({
  selector: 'app-root',
  standalone: true,
  template: `
  <button (click)="onClick(7)">Speeder</button>
  <button (click)="onClick(8)">Starfighter</button>
  <div>Name: {{ product.value()?.name }}</div>
  <div>Cost: {{ product.value()?.cost_in_credits }}</div>
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);
  color = signal('slate gray');

  // This would normally reside in a service
//   products = resource({
//    loader: () => fetch(this.url)
//                .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
```

NG9: Property 'onClick' does not exist on type 'App'.

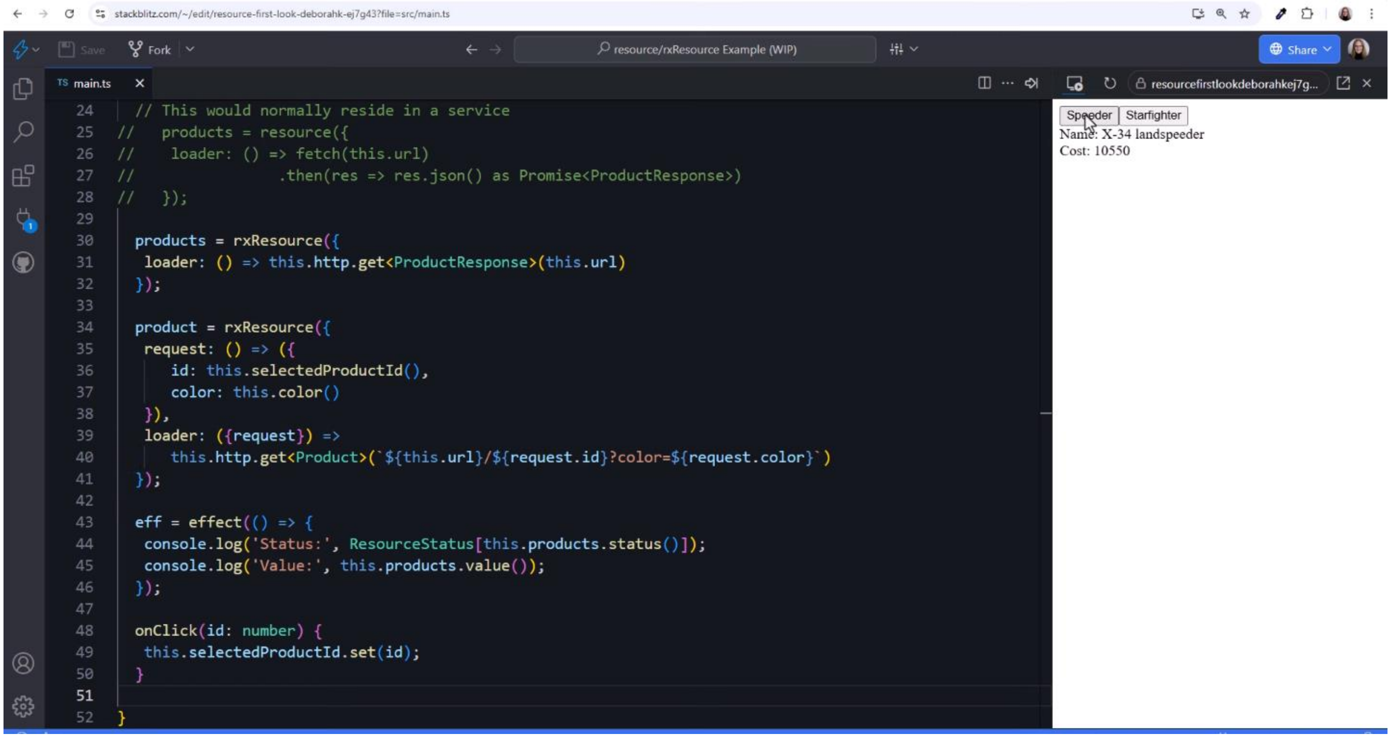
src/main.ts:11:19

Click outside, press Esc key, or fix the code to dismiss.


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter

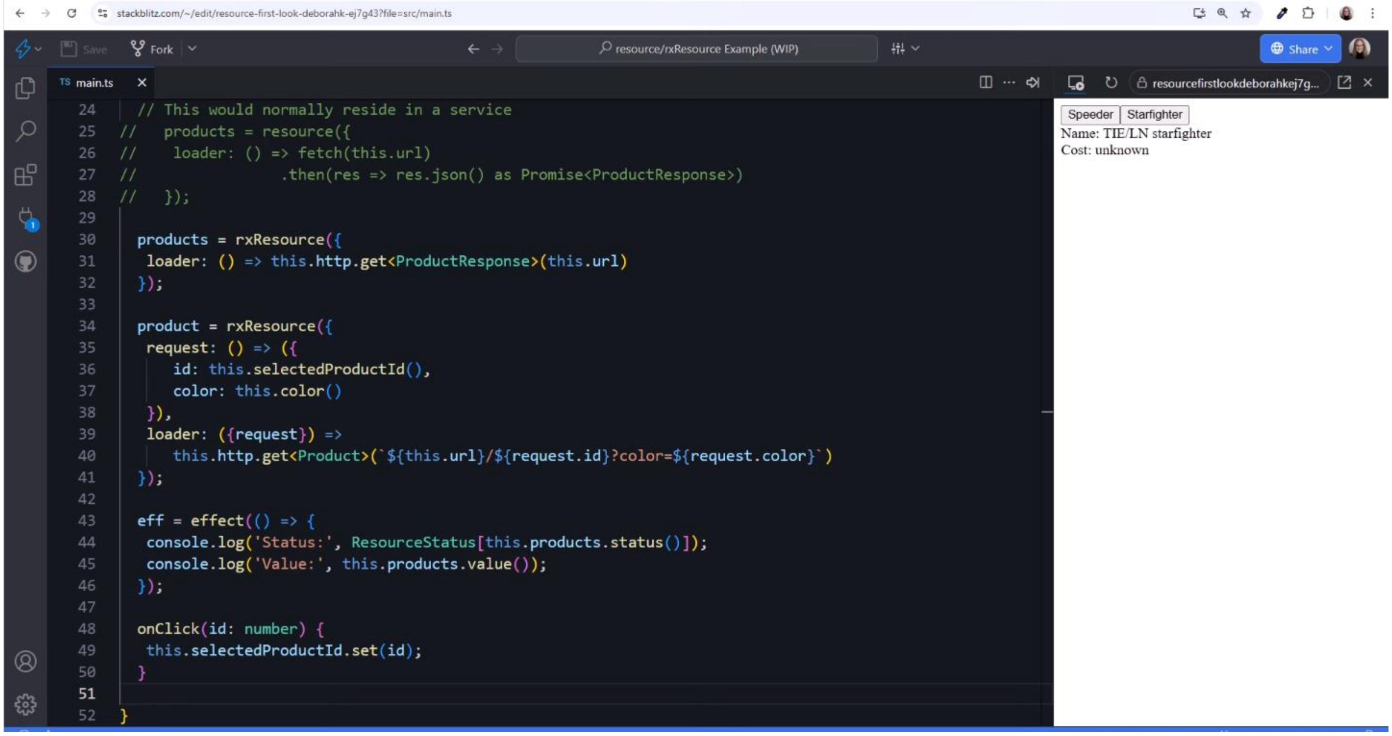
```
  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

}
```

```
Name:
Cost:
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter

```
  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

}
```

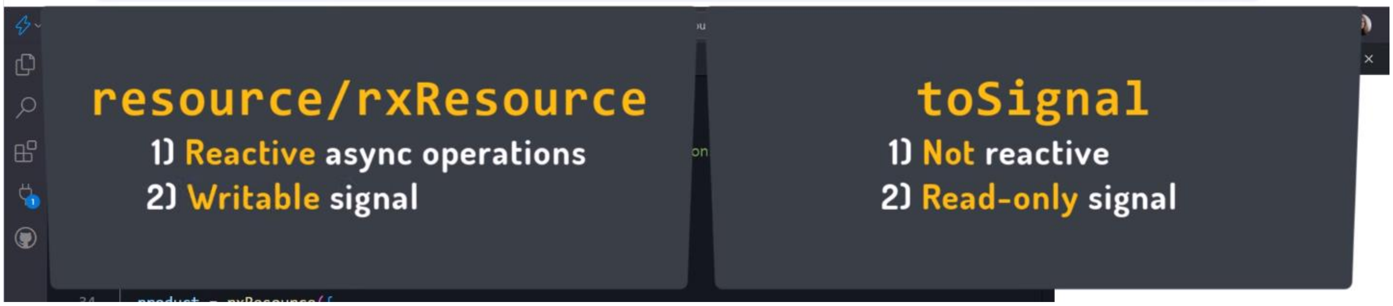
```
Name: X-34 landspeeder
Cost: 10550
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.products.status()]);
   console.log('Value:', this.products.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

}
```

```
Speeder Starfighter
Name: TIE/LN starfighter
Cost: unknown
```

It works too. Let us update the price from **undefine**



resource/rxResource

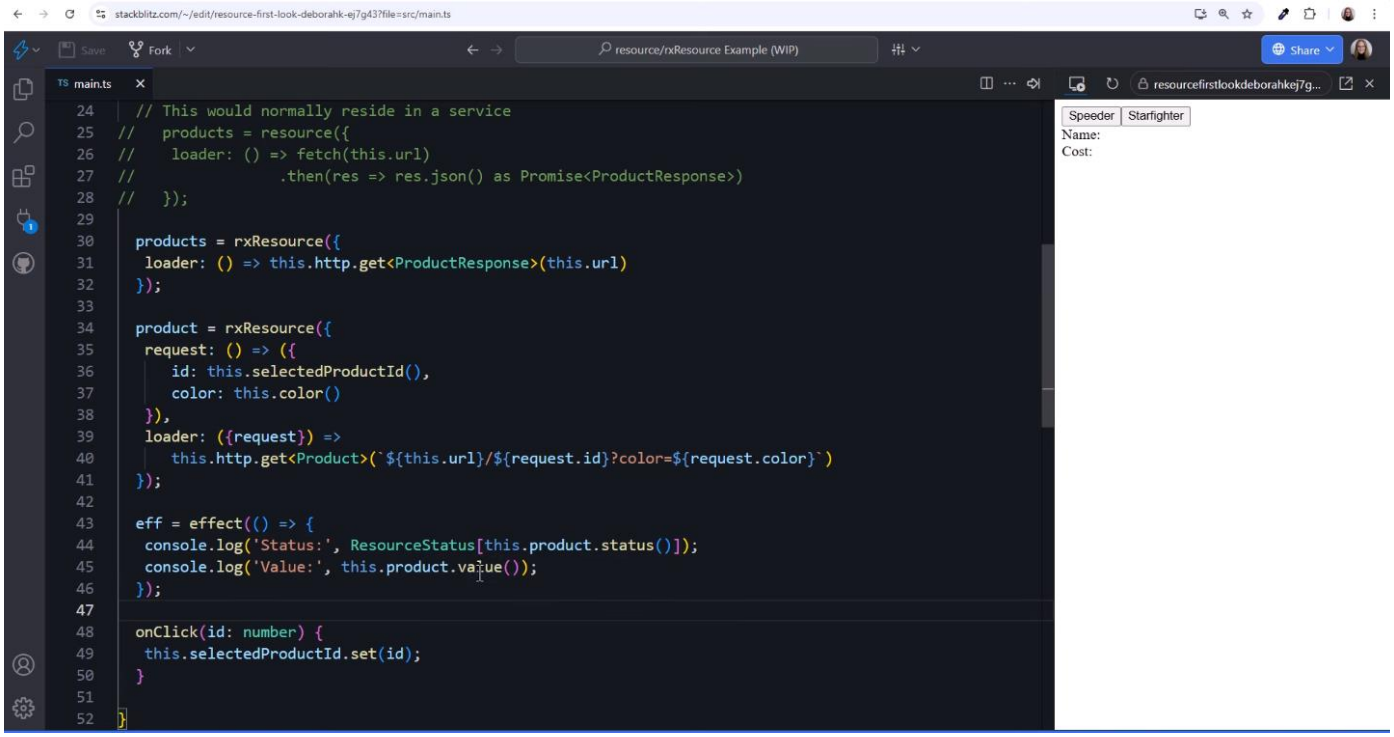
1) Reactive async operations

2) Writable signal

toSignal

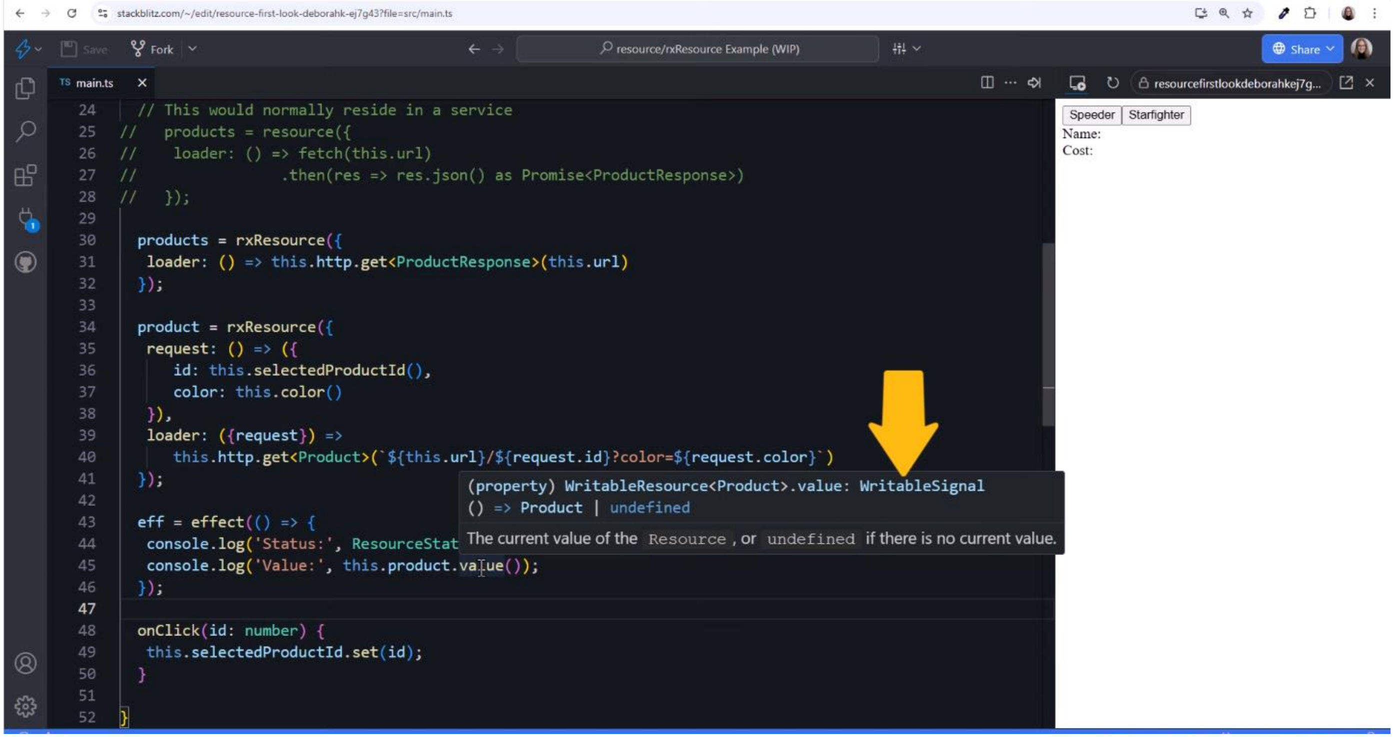
1) Not reactive

2) Read-only signal


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...

```
  // This would normally reside in a service
//   products = resource({
//     loader: () => fetch(this.url)
//                 .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.product.status()]);
   console.log('Value:', this.product.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

}
```

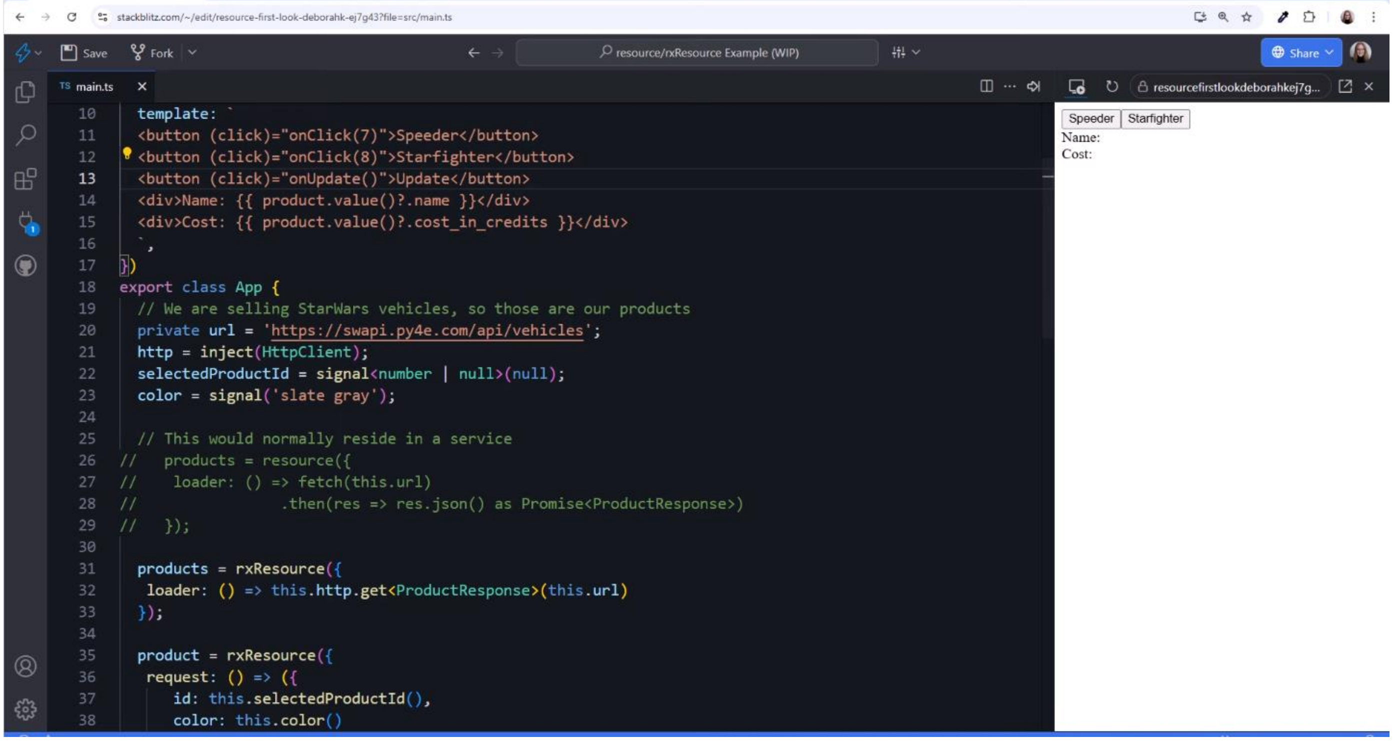
```
Speeder Starfighter
Name:
Cost:
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter
Name:
Cost:

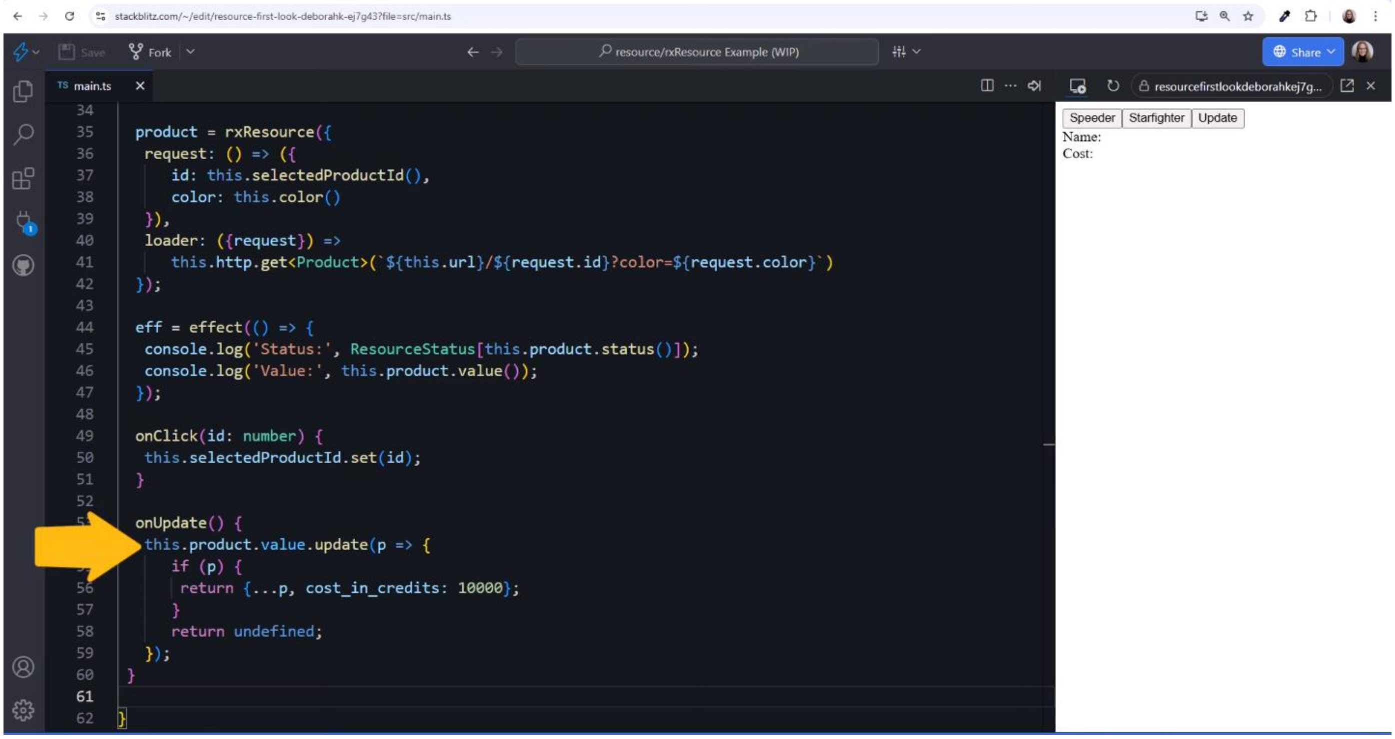
```
  // This would normally reside in a service
//   products = resource({
//    loader: () => fetch(this.url)
//                .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStat
   console.log('Value:', this.product.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

}
```

```
(property) WritableResource<Product>.value: WritableSignal
() => Product | undefined
```

The current value of the `Resource` , or `undefined` if there is no current value.

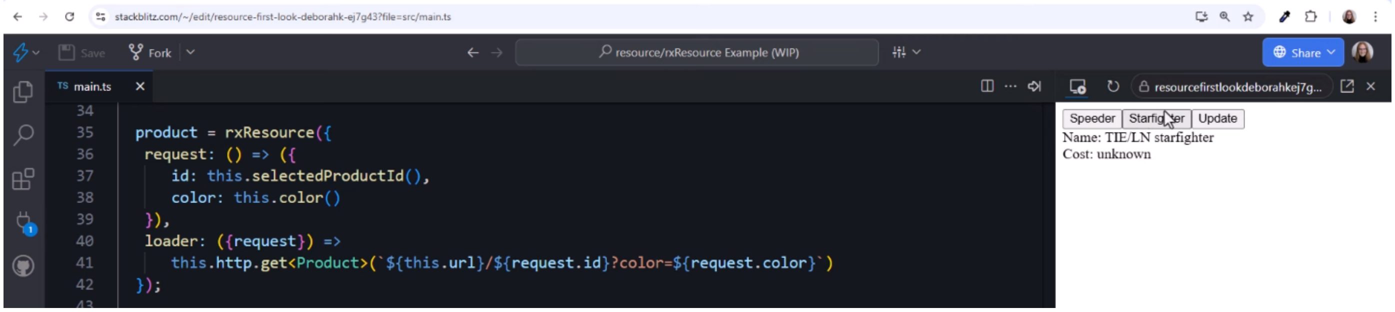

stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter
Name:
Cost:

```
  template: `
  <button (click)="onClick(7)">Speeder</button>
  <button (click)="onClick(8)">Starfighter</button>
  <button (click)="onUpdate()">Update</button>
  <div>Name: {{ product.value()?.name }}</div>
  <div>Cost: {{ product.value()?.cost_in_credits }}</div>
  `,
})
export class App {
  // We are selling StarWars vehicles, so those are our products
  private url = 'https://swapi.py4e.com/api/vehicles';
  http = inject(HttpClient);
  selectedProductId = signal<number | null>(null);
  color = signal('slate gray');

  // This would normally reside in a service
//   products = resource({
//    loader: () => fetch(this.url)
//                .then(res => res.json() as Promise<ProductResponse>)
//   });

  products = rxResource({
   loader: () => this.http.get<ProductResponse>(this.url)
  });

  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
```


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter
Update

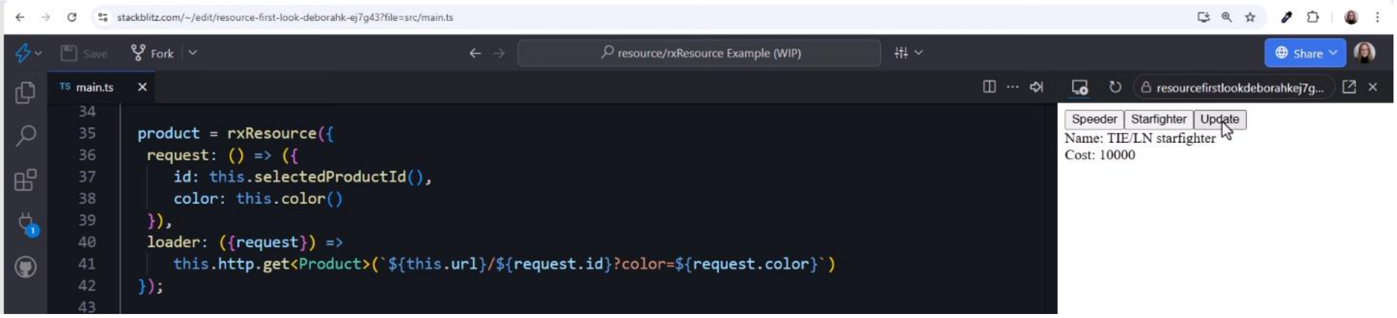
```
  product = rxResource({
   request: () => ({
     id: this.selectedProductId(),
     color: this.color()
   }),
   loader: ({request}) =>
     this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.product.status()]);
   console.log('Value:', this.product.value());
  });

  onClick(id: number) {
   this.selectedProductId.set(id);
  }

  onUpdate() {
   this.product.value.update(p => {
     if (p) {
      return {...p, cost_in_credits: 10000};
     }
     return undefined;
   });
  }
}
```

Name:
Cost:


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter
Update

```
  product = rxResource({
   request: () => ({
     id: this.selectedProductId(),
     color: this.color()
   }),
   loader: ({request}) =>
     this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });
```

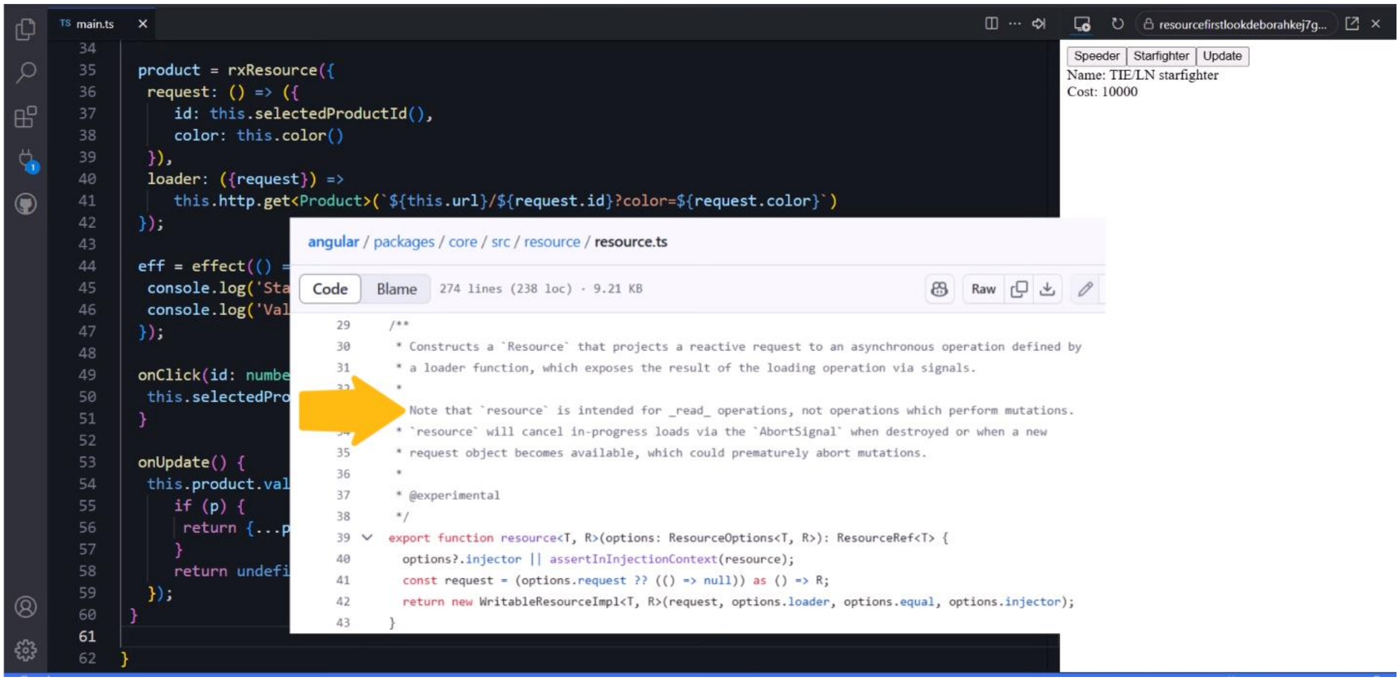
Name: TIE/LN starfighter
Cost: unknown


stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
resource/rxResource Example (WIP)
Share
main.ts
resourcefirstlookdeborahkej7g...
Speeder
Starfighter
Update

```
  product = rxResource({
   request: () => ({
     id: this.selectedProductId(),
     color: this.color()
   }),
   loader: ({request}) =>
     this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });
```

Name: TIE/LN starfighter
Cost: 10000

The value is now updated, we could post the updated value back to the server but it is only for read only.



Name: TIE/LN starfighter
Cost: 10000

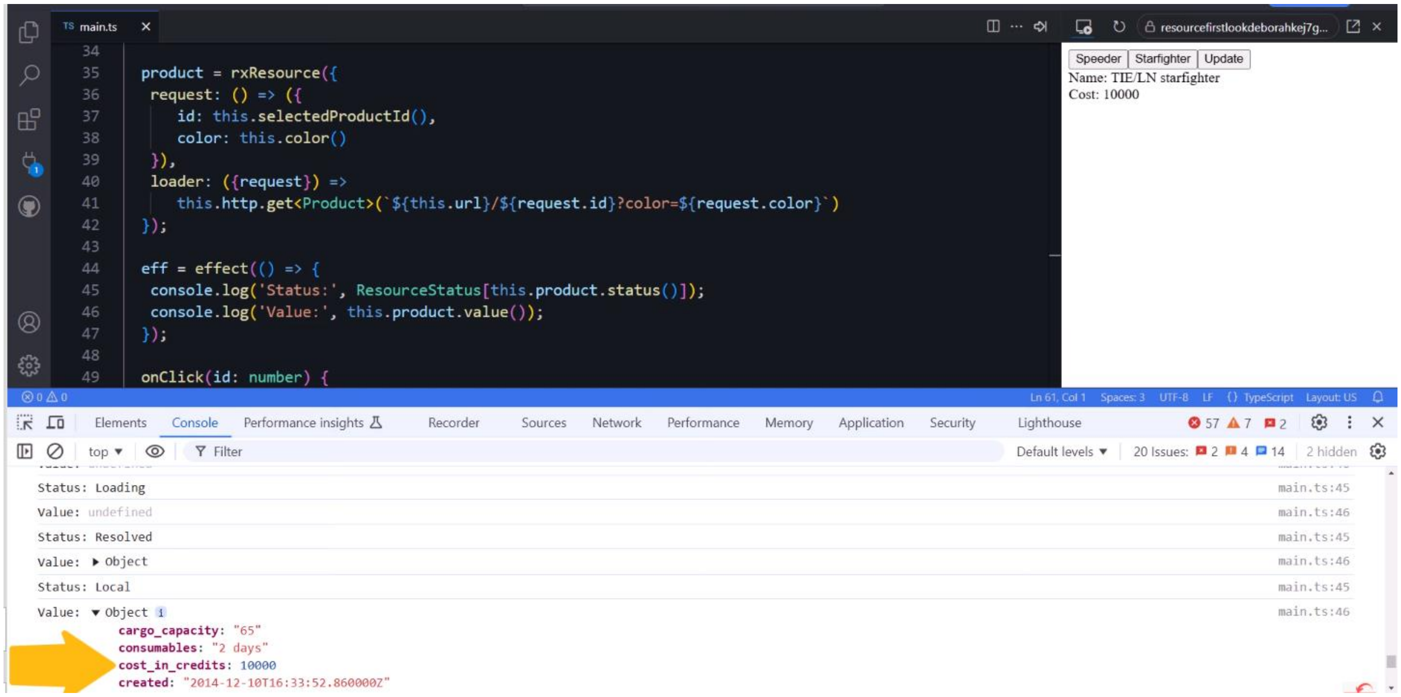
```
  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() =
   console.log('Sta
   console.log('Val
  });

  onClick(id: numbe
   this.selectedPro
  }

  onUpdate() {
   this.product.val
      if (p) {
       return {...p
      }
      return undefi
   });
  }
}
```

angular / packages / core / src / resource / **resource.ts**

274 lines (238 loc) · 9.21 KB

```
/**
 * Constructs a `Resource` that projects a reactive request to an asynchronous operation defined by
 * a loader function, which exposes the result of the loading operation via signals.
 *
 * Note that `resource` is intended for _read_ operations, not operations which perform mutations.
 * `resource` will cancel in-progress loads via the `AbortSignal` when destroyed or when a new
 * request object becomes available, which could prematurely abort mutations.
 *
 * @experimental
 */
export function resource<T, R>(options: ResourceOptions<T, R>): ResourceRef<T> {
  options?.injector || assertInInjectionContext(resource);
  const request = (options.request ?? (() => null)) as () => R;
  return new WritableResourceImpl<T, R>(request, options.loader, options.equal, options.injector);
}
```

Don’t use **resource()** or **rxResource()** for **PUT**, **POST** or **DELETE** HTTP operations, only for the GET operation.

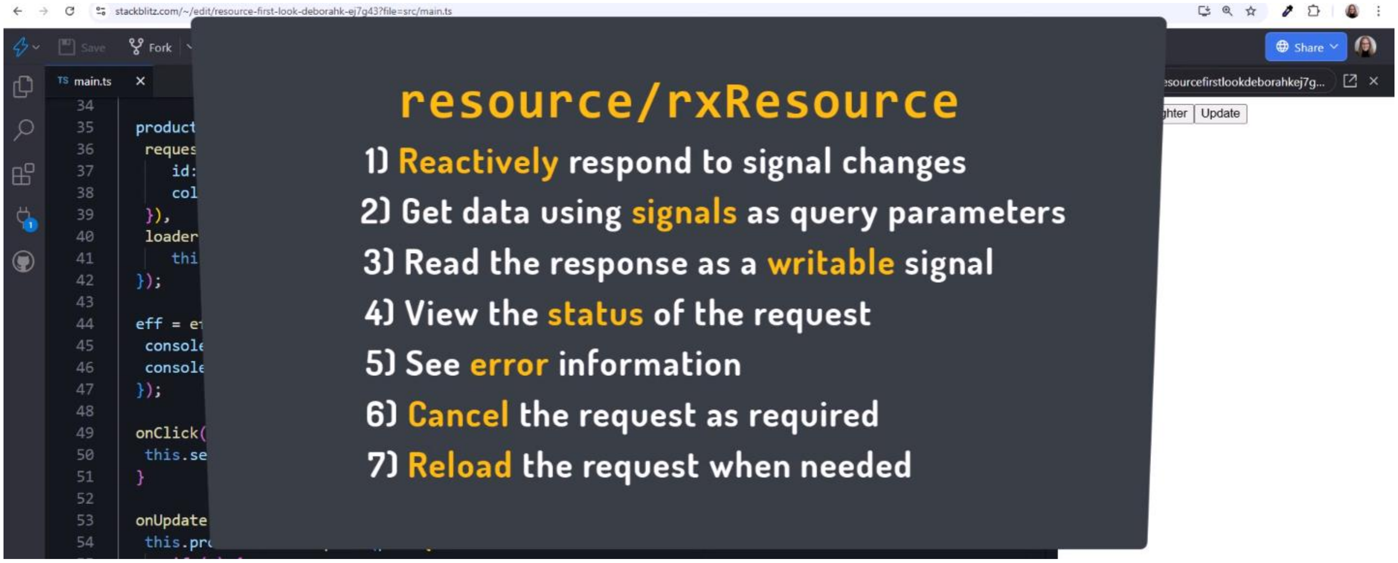


Name: TIE/LN starfighter
Cost: 10000

```
  product = rxResource({
   request: () => ({
      id: this.selectedProductId(),
      color: this.color()
   }),
   loader: ({request}) =>
      this.http.get<Product>(`${this.url}/${request.id}?color=${request.color}`)
  });

  eff = effect(() => {
   console.log('Status:', ResourceStatus[this.product.status()]);
   console.log('Value:', this.product.value());
  });

  onClick(id: number) {
```

```
Status: Loading                                         main.ts:45
Value: undefined                                        main.ts:46
Status: Resolved                                        main.ts:45
Value: ▶ Object                                         main.ts:46
Status: Local                                           main.ts:45
Value: ▼ Object i                                       main.ts:46
          cargo_capacity: "65"
          consumables: "2 days"
          cost_in_credits: 10000
          created: "2014-12-10T16:33:52.860000Z"
```

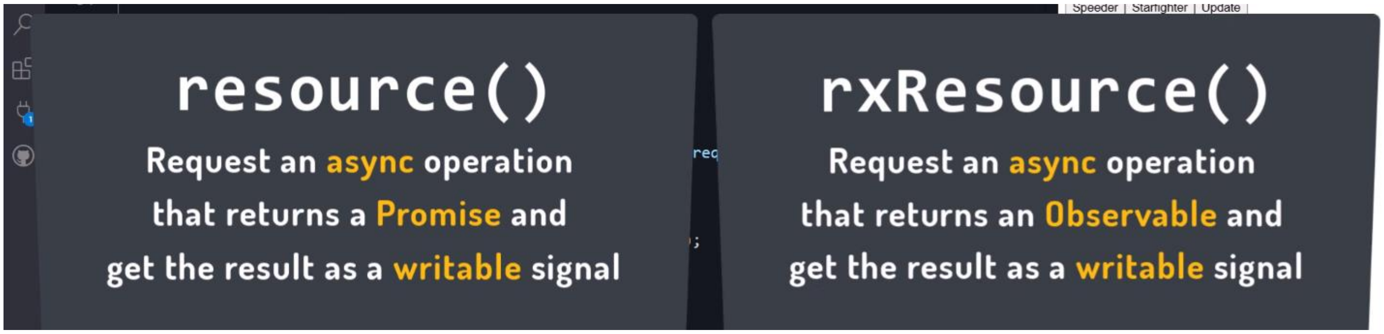
This is the update for the data locally.



stackblitz.com/~/edit/resource-first-look-deborahk-ej7g43?file=src/main.ts
Save
Fork
main.ts
Share
Update

## resource/rxResource

1) Reactively respond to signal changes
2) Get data using signals as query parameters
3) Read the response as a writable signal
4) View the status of the request
5) See error information
6) Cancel the request as required
7) Reload the request when needed



## resource()

Request an async operation that returns a Promise and get the result as a writable signal

## rxResource()

Request an async operation that returns an Observable and get the result as a writable signal

These features will be experimental in Angular v19 in Nov 2024.