


# export

 Sprachen

Das **export**-Statement wird verwendet um funktionen und Objekte aus einer gegebenen Datei (oder *Modul*) zu exportieren.

## Syntax

```
export { name1, name2, ..., nameN };
export { variable1 as name1, variable2 as name2, ..., nameN };
export let name1, name2, ..., nameN; // oder: var
export let name1 = ..., name2 = ..., ..., nameN; // oder: var, const

export default expression;
export default function (...) { ... } // oder: class, function*
export default function name1(...) { ... } // oder: class, function*
export { name1 as default, ... };

export * from ...;
export { name1, name2, ..., nameN } from ...;
export { import1 as name1, import2 as name2, ..., nameN } from ...;
```

### nameN

Bezeichner der exportiert werden soll (damit er in einem anderen Script via [import](#) importiert werden kann).

# Beschreibung

Es gibt zwei verschiedene Arten von Exports, die jeweils der oben angegebene Syntax entsprechen:

- Benannte Exports:

```
1 | export { myFunction }; // exportiert eine Funktion, die zuv  
2 | export const foo = Math.sqrt(2); // exportiert eine Konstan
```

- Default-Exports (nur einer je Script):

```
1 | export default function() {} // oder 'export default class  
2 | // hier ist kein Semikolon
```

Benannte Exports sind nützlich um mehrere Werte zu exportieren. Beim Import kann man den selben Namen verwenden um auf den entsprechenden Wert zu verweisen.

Bezüglich Default-Export: es kann nur einen einzigen Default-Export pro Modul geben. Ein Default-Export kann eine Funktion sein, eine Klasse, ein Objekt oder irgendetwas anderes. Da dieser Wert am einfachsten importiert werden kann wird er als der "Haupt-Export" des Moduls angesehen.

---

## Beispiele

### Benannte Exports

Im Modul können wir den folgenden Code verwenden:

```
1 | // Modul "my-module.js"  
2 | function cube(x) {  
3 |     return x * x * x;  
4 | }  
5 | const foo = Math.PI + Math.SQRT2;  
6 | export { cube, foo };
```

Daraufhin könnten wir nun in einem anderen Script (cf. [import](#)) wie folgt vorgehen:

```
1 | import { cube, foo } from 'my-module';
2 | console.log(cube(3)); // 27
3 | console.log(foo);      // 4.555806215962888
```

## Standard-Export

Wenn wir nur einen einzelnen Wert exportieren wollen, oder einen Fallback-Wert für unser Modul zur Verfügung haben möchten, können wir einen Default-Export verwenden:

```
1 | // Modul "my-module.js"
2 | export default function cube(x) {
3 |     return x * x * x;
4 | }
```

In einem anderen Script kann dieser Default-Export dann unkompliziert importiert werden:

```
1 | import myFunction from 'my-module';
2 | console.log(myFunction(3)); // 27
```




























---

## Spezifikationen

Spezifikation	Status	Kommentar
<a href="#">ECMAScript 2015 (6th Edition, ECMA-262)</a> Die Definition von 'Exports' in dieser Spezifikation.	<b>ST</b> Standard	Initiale Definition.
<a href="#">ECMAScript Latest Draft (ECMA-262)</a> Die Definition von 'Exports' in dieser Spezifikation.	<b>D</b> Entwurf	

---

## Browserkompatibilität

Grundlegende Unterstützung		
 	61	
 	16	
 	60	
 	Nein	
 	47	
 	10.1	
 	Nein	
 	61	
 	Ja	
 	60	
 	47	
 	10.1	
 	Nein	
	?	



Vollständige Unterstützung



Keine Unterstützung



Compatibility unknown



Benutzer muss dieses Feature explizit aktivieren.

## Siehe auch

- [import](#)
- [ES6 in Depth: Modules](#), Hacks Blog-Post von Jason Orendorff
- [Axel Rauschmayer's Buch: "Exploring JS: Modules"](#)

---

🔍 Schlagwörter: [ECMAScript 2015](#) [export](#) [JavaScript](#) [Modules](#) [Statement](#)

👤 Mitwirkende an dieser Seite: [xchange11](#), [schlagi123](#), [Snapstromegon](#), [thomaskempel](#), [yampus](#), [roehrig](#), [tuffi111](#), [sbusch](#)

🕒 Zuletzt aktualisiert von: [xchange11](#), 06.04.2018, 03:32:34

---

[Webtechnologien für Entwickler](#) > [JavaScript](#) > [JavaScript-Referenz](#) > [Anweisungen und Deklarationen](#) > [export](#)

---

## Verwandte Themen

### *JavaScript*

#### Tutorials:

- ▶ [Einleitend](#)
- ▶ [JavaScript Guide](#)
- ▶ [Fortgeschritten](#)
- ▶ [Erweitert](#)

#### Referenzen:

- ▶ [Standardobjekte](#)
- ▶ [Ausdrücke & Operatoren](#)
- ▼ [Anweisungen & Deklarationen](#)

[Legacy generator function](#) [\[Übersetzen\]](#)

[async function](#) [\[Übersetzen\]](#)

[block](#)

[break](#)

[Klasse](#)

[const](#)

[continue](#)

[debugger](#) [\[Übersetzen\]](#)

[default](#)

[do...while](#)

[empty](#)

[export](#)

[for](#)

👤 🗑️ [for each...in](#)

[for...in](#)

[for...of](#)

Funktion

function\*

if...else

import

label

let

return

switch

throw

try...catch

var

while

 **with** [Übersetzen]

- ▶ Funktionen
- ▶ Klassen
- ▶ Fehler
- ▶ Weiteres
- ▶ Neu in JavaScript

#### Dokumentation:

- ▶ Nützliche Listen
- ▶ Mitmachen

---

# Lernen Sie das Beste aus dem Bereich Web-Entwicklung

×

Erhalten Sie das Neueste und Wichtigste von MDN direkt in Ihren Posteingang.

*Der Newsletter wird derzeit nur auf Englisch angeboten.*

Melden Sie sich jetzt an



MDN web docs  
moz://a

[Web-Technologien](#)

[Lernen Sie Web-Entwicklung](#)

[Über MDN](#)

[Rückmeldung](#)



---

**moz://a**

[Über](#)

[Kontakt](#)

[Firefox](#)



---

Weitere Sprachen: Deutsch (de) ▼

---

[Nutzungsbedingungen](#) [Datenschutz](#) [Cookies](#)

© 2005–2018 Mozilla und einzelne Mitwirkende. Inhalt steht unter [diesen Lizenzen](#).