
```
"Hello Topiconf"
```


```
let greeting_var: string;

greeting_var = "Hello Topiconf";
```


```
let greeting_var: "Hello Topiconf";

greeting_var = "Hello Topiconf";
```


```
let greeting_var: "Hello Topiconf"
                        | "Hello world";
greeting_var = "Hello Topiconf";
```


```
let greeting_var: `Hello
                  ${"Topiconf"|"world"}`;
greeting_var = "Hello Topiconf";
```

who am i

- Sander Evers
- software engineer
- Topicus Education (2012)
- CumLaude team (2021)
- becoming a TypeScript fanboi

js improvements over 10 years

```
[1,2,3].map(x => 10*x)
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)
queryParams = {...defaultParams, max:100}
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)
queryParams = {...defaultParams, max:100}
data = fetch().then(resp => parse(resp))
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
while (true)
  yield i++
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
while (true)
  yield i++

item?.title?.toUpper() ?? 'Untitled'
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
while (true)
  yield i++

item?.title?.toUpper() ?? 'Untitled'

`${n} bottles of ${item} on the wall`
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
while (true)
  yield i++

item?.title?.toUpper() ?? 'Untitled'

`${n} bottles of ${item} on the wall`

import / export
```

js improvements over 10 years

```
[1,2,3].map(x => 10*x)

queryParams = {...defaultParams, max:100}

data = fetch().then(resp => parse(resp))

while (true)
  item = await queue.read()
while (true)
  yield i++

item?.title?.toUpper() ?? 'Untitled'

`${n} bottles of ${item} on the wall`

import / export

class TodoList extends Component {}
```

```
# but still a crappy standard library
```

```
[1,5,10].sort()
```

```
_.sortBy([1,5,10])
```

TypeScript = JavaScript + type annotations

```
type Fraction = {num: number, den: number}

function multiply(a: Fraction, b: Fraction): Fraction {
  return {
    num: a.num * b.num,
    den: a.den * b.den
  };
}
```

why typing

- QA: prove absence of errors
- documentation
- maintenance / refactoring
- code completion

```ts
type SimpleNounPhrase = `${Particle} ${Noun}`;

type Particle = 'de' | 'het' | 'een';

type Noun = FM | N;
type FM = 'boom' | 'roos' | 'vis';
type N = 'huis' | 'raam';
```

```ts
type SimpleNounPhrase = `${Particle} ${Noun}`;

type Particle = 'de' | 'het' | 'een';

type Noun = FM | N;
type FM = 'boom' | 'roos' | 'vis';
type N = 'huis' | 'raam';

const ex1: SimpleNounPhrase = 'raam het';
```
❌

```ts
type SimpleNounPhrase = `${Particle} ${Noun}`;

type Particle = 'de' | 'het' | 'een';

type Noun = FM | N;
type FM = 'boom' | 'roos' | 'vis';
type N = 'huis' | 'raam';

const ex1: SimpleNounPhrase = 'raam het'; ❌
const ex2: SimpleNounPhrase = 'het raam'; ✅
```

```ts
type SimpleNounPhrase = `${Particle} ${Noun}`;

type Particle = 'de' | 'het' | 'een';

type Noun = FM | N;
type FM = 'boom' | 'roos' | 'vis';
type N = 'huis' | 'raam';

const ex1: SimpleNounPhrase = 'raam het'; ❌
const ex2: SimpleNounPhrase = 'het raam'; ✅
const ex3: SimpleNounPhrase = 'de raam';  ✅
```

```
type ParticleMapping = { de: FM; het: N; een: FM | N };
```

```ts
type ParticleMapping = { de: FM; het: N; een: FM | N };

const ex4: ParticleMapping = {
    de: 'boom',
    het: 'huis',
    een: 'vis'
}
```

```ts
type ParticleMapping = { de: FM; het: N; een: FM | N };

const ex4: ParticleMapping = {
    de: 'boom',
    het: 'huis',
    een: 'vis'
}

type CorrectMap = { [p in Particle]: `${p} ${ParticleMapping[p]}` };
```

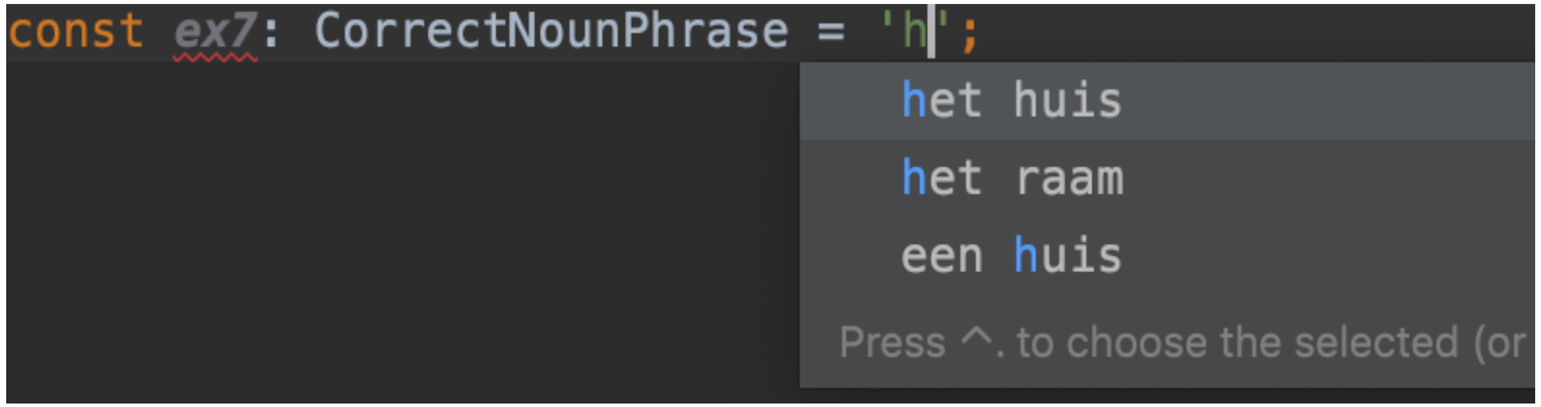
```ts
type ParticleMapping = { de: FM; het: N; een: FM | N };

const ex4: ParticleMapping = {
    de: 'boom',
    het: 'huis',
    een: 'vis'
}

type CorrectMap = { [p in Particle]: `${p} ${ParticleMapping[p]}` };

 ==>   { de: `de ${FM}`; het: `het ${N}`; een: `een ${FM | N}`}
```

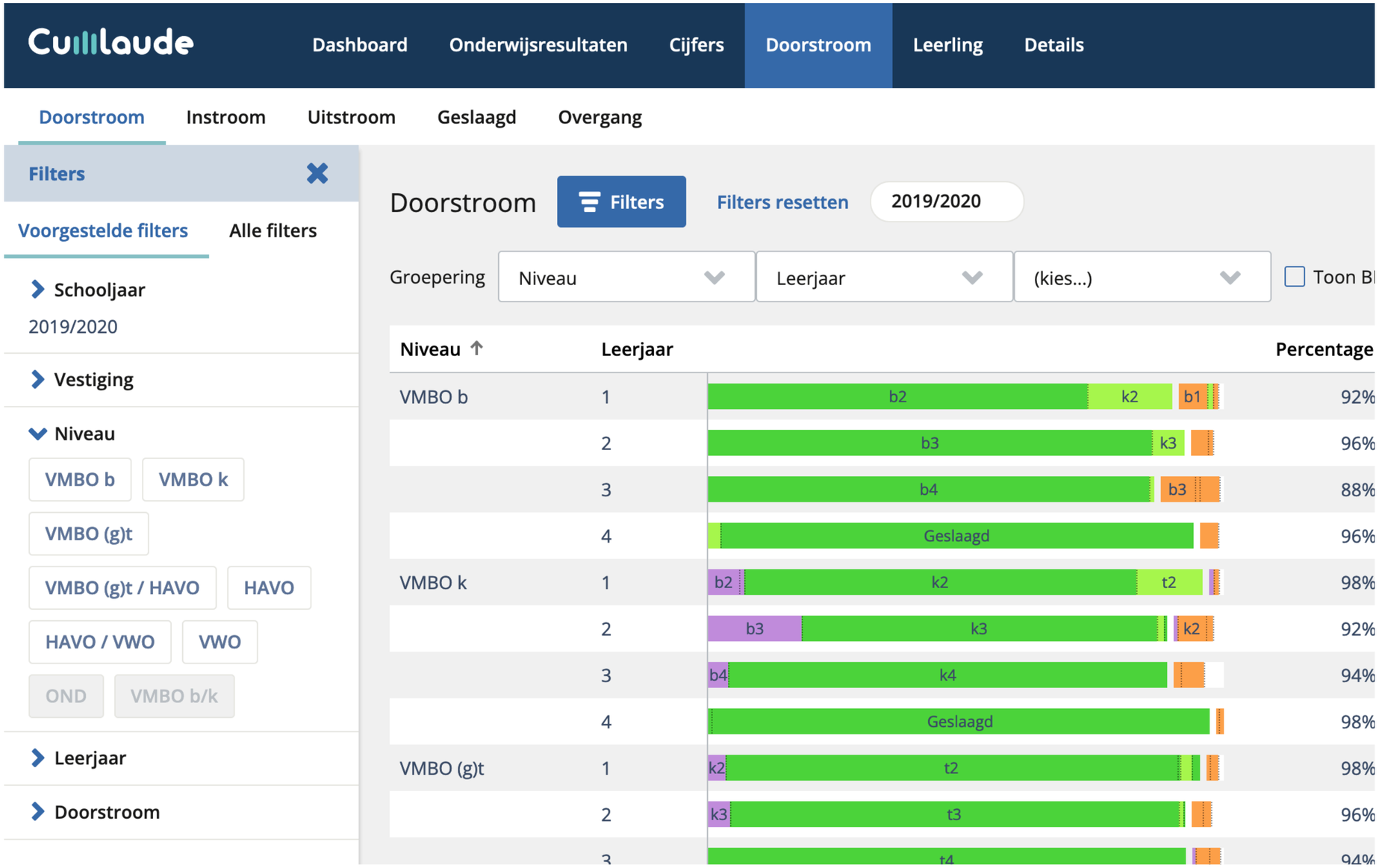
```ts
type ParticleMapping = { de: FM; het: N; een: FM | N };

const ex4: ParticleMapping = {
    de: 'boom',
    het: 'huis',
    een: 'vis'
}

type CorrectMap = { [p in Particle]: `${p} ${ParticleMapping[p]}` };

 ==>   { de: `de ${FM}`; het: `het ${N}`; een: `een ${FM | N}`}



const ex5: CorrectMap = {
    de: 'de roos',
    het: 'het raam',
    een: 'een huis'
}
```

```
type CorrectNounPhrase = CorrectMap[keyof CorrectMap];
```

```ts
type CorrectNounPhrase = CorrectMap[keyof CorrectMap];

const ex6: CorrectNounPhrase = 'de raam';  ❌
const ex7: CorrectNounPhrase = 'het raam'; ✅
```

```ts
type CorrectNounPhrase = CorrectMap[keyof CorrectMap];

const ex6: CorrectNounPhrase = 'de raam';   ❌
const ex7: CorrectNounPhrase = 'het raam';  ✅
```



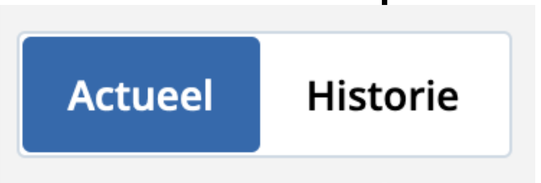

Cumlaude
Dashboard
Onderwijsresultaten
Cijfers
Doorstroom
Leerling
Details
Doorstroom
Instroom
Uitstroom
Geslaagd
Overgang
Filters
Voorgestelde filters
Alle filters
Schooljaar
2019/2020
Vestiging
Niveau
VMBO b
VMBO k
VMBO (g)t
VMBO (g)t / HAVO
HAVO
HAVO / VWO
VWO
OND
VMBO b/k
Leerjaar
Doorstroom
Doorstroom
Filters
Filters resetten
2019/2020
Groepering
Niveau
Leerjaar
(kies...)
Toon B
Niveau
Leerjaar
Percentage
VMBO b 1 b2 k2 b1 92%
2 b3 k3 96%
3 b4 b3 88%
4 Geslaagd 96%
VMBO k 1 b2 k2 t2 98%
2 b3 k3 k2 92%
3 b4 k4 94%
4 Geslaagd 98%
VMBO (g)t 1 k2 t2 98%
2 k3 t3 96%
3 t4 94%

```
interface FacLbLoopbaan {
    lb_d_inschrijving?: string;
    lb_d_uitschrijving?: string;
    lb_fk_br_vest?: DimBrBrin;
    lb_fk_br_vest_volgend_pl?: DimBrBrin;
    lb_fk_ilt?: DimIltOpleidingscode;
    lb_fk_ilt_volgend_pl?: DimIltOpleidingscode;
    ...
}

interface DimIltOpleidingscode {
    ilt_abb_profiel?: string;
    ilt_co_niveau?: number;
    ilt_nm_niveau?: Niveau;
    ...
}
```

```
interface FacLbLoopbaan {
    lb_d_inschrijving?: string;
    lb_d_uitschrijving?: string;
    lb_fk_br_vest?: DimBrBrin;
    lb_fk_br_vest_volgend_pl?: DimBrBrin;
    lb_fk_ilt?: DimIltOpleidingscode;
    lb_fk_ilt_volgend_pl?: DimIltOpleidingscode;
    ...
}

interface DimIltOpleidingscode {
    ilt_abb_profiel?: string;
    ilt_co_niveau?: number;
    ilt_nm_niveau?: Niveau;
    ...
}

doorstroomFilters: FilterName[] = [
  'x_schooljaar_historie',
  'x_multiselect_schooljaar',
  'lb_nm_vestiging',
  'lb_fk_ilt.ilt_nm_niveau',
  'lb_nm_leerjaar',
  'lb_nm_idu',
];
```

doorstroom.component.html

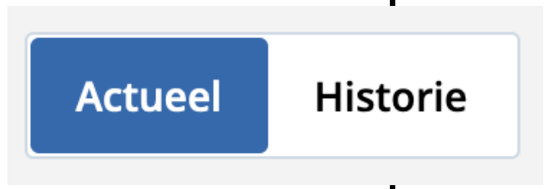
```
...

<app-toggle-buttons class="right"
  [value]="variant"
  (valueChange)="qp.dispatch('variant',$event)">
</app-toggle-buttons>
```



doorstroom.component.html

```
...

<app-toggle-buttons class="right"
  [value]="variant"
  (valueChange)="qp.dispatch('variant',$event)">
</app-toggle-buttons>
```

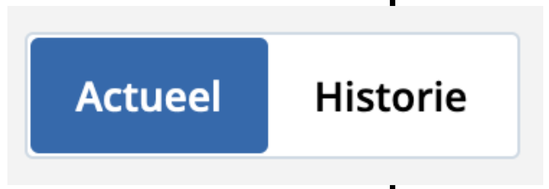
```
enum DashboardVariant
```



doorstroom.component.html

```
...

<app-toggle-buttons class="right"
  [value]="variant"
  (valueChange)="qp.dispatch('variant',$event)">
</app-toggle-buttons>
```

```
enum DashboardVariant
```



query-param.service.ts

```
const configs = {
  adviesType: attrPathCodec,
  'brin-overgang': booleanCodec,
  col: multiCodec<CijferColumn>(),
  threshold: numberCodec,
  variant: singleCodec<DashboardVariant>(),
};

dispatch<T extends keyof typeof configs>
  (paramName: T, value: Parameters<typeof configs[T]['encode']>[0])
{
  ...
}
```

"Thank you Topiconf"