

Grafieken

applicatiebouw

OpenWave heeft de open source library charts.js geïmporteerd, waardoor het mogelijk is tellingen op de database te laten zien als lijn-, staaf-, punten- of taartdiagram.

Dit kan vooralsnog alleen op een tegel.

In plaats van een tegel live te vullen met tekst op basis van een query (zie [getTileContent](#) op [Portaltegel](#)), kan nu ook de tegel live gevuld worden met een diagram op basis van een query.

De uitkomst van die query moet een JSON-string zijn met daarin opgenomen labels en metadata en data van het diagram, zoals de charts.js library deze voorschrijft. Zie <https://github.com/chartjs/Chart.js>

voorbeeld staafdiagram



Bovenstaande tegel toont de aantallen zaken van de afgelopen 4 jaar in een staafdiagram. Daartoe moet:

- de tegelcontent verwijzen naar `getflexChart()` met een querynaam als parameter
- en moet een passende query gedefinieerd worden onder deze querynaam.

getFlexChart()

Op het detailscherm van de tegeldefinitie (beheerportaal: kolom *Scher- en Tegelbeheer* : tegel *Portal*) van de betreffende tegel moet de kolom *Tegelopschrift dynamisch met API* `getTileContent()` of `getFlexChart()` (`dvgettilecontent`) gevuld te worden met een aanroep naar `getFlexChart` met twee parameters: de gewenste diagramsoort en de naam van de query die de json string definieert.

De functie `getFlexChart` evalueert de query en geeft het resultaat met het gewenste diagramtype door aan de `chart.js`, die de informatie rendert naar een diagram.

De diagramsoort kan zijn: `pie`, `bar`, of `line`. In bovenstaand voorbeeld is dat `bar`.

Dus de waarde van de kolom `dvgettilecontent` in de tegeldefinitie van bovenstaand voorbeeld is `getFlexChart(bar,opening_aantallenperjaar)` waarbij `opening_aantallenperjaar` de naam van de query is (`tbquery.dvcode`) die een jsonstring met de nodige informatie genereert. (OpenWave levert deze queries niet standaard uit:U dient deze zelf te definiëren)

query

De `char.js` library verwacht een json string met daarin metadata (zoals labels en kleuren) en data (aantallen). Bij een bar (staafdiagram) ziet zo'n JSON-string er (naar bovenstaand voorbeeld) zo uit:

```
{
  "labels": ["2021", "2022", "2023", "2024"],
  "datasets": [{
    "label": "Aantal zaken per jaar",
    "backgroundColor": "rgba(0, 255, 255, 1)",
    "borderColor": "rgba(255, 0, 0, 1)",
    "borderWidth": 0.5,
    "data": [11230, 8578, 12512, 2200]
  }]
}
```

```
}
```

De uitkomst van de query moet dus bovenstaande string opleveren waarbij de labels berekend moeten worden (de laatste 4 jaar) en waarbij per jaar de aantallen zaken (de data) opgehaald moeten worden uit de database.

```
select '{
"labels": ["' || (select extract('year'from current_date)-3 ||',' ||
              extract('year'from current_date)-2 ||',' ||
              extract('year'from current_date)-1 ||',' ||
              extract('year'from current_date))||' "'
          || ']',
"datasets": [{
"label":"Aantal zaken per jaar",
"backgroundColor": "rgba(0, 255, 255, 1)",
"borderColor": "rgba(255, 0, 0, 1)",
"borderWidth": 0.5,
"data": [' ||
          (select count(*) from vwfrmalleaanvragen where ddaanvraag
>= to_date('0101' ||extract('year'from current_date)-3,'ddmmyyyy') and
ddaanvraag <= to_date('3112' ||extract('year'from
current_date)-3,'ddmmyyyy')) || ',' ||
          (select count(*) from vwfrmalleaanvragen where ddaanvraag
>= to_date('0101' ||extract('year'from current_date)-2,'ddmmyyyy') and
ddaanvraag <= to_date('3112' ||extract('year'from
current_date)-2,'ddmmyyyy')) || ',' ||
          (select count(*) from vwfrmalleaanvragen where ddaanvraag
>= to_date('0101' ||extract('year'from current_date)-1,'ddmmyyyy') and
ddaanvraag <= to_date('3112' ||extract('year'from
current_date)-1,'ddmmyyyy')) || ',' ||
          (select count(*) from vwfrmalleaanvragen where ddaanvraag
>= to_date('0101' ||extract('year'from current_date),'ddmmyyyy') and
ddaanvraag <= to_date('3112' ||extract('year'from current_date),'ddmmyyyy'))
          || ']'
        ]}]
}'
```

voorbeeld taartdiagram



Bovenstaande tegel toont de de verdeling van de openstaande omgevingzaken per soort (dvsoortproc van tbsoortomgzaak).

Daartoe moet:

- de tegelcontent verwijzen naar `getflexChart()`
- en moet een passende query gedefinieerd worden.

getFlexChart()

De waarde van de kolom `dvgettilecontent` in bovenstaand voorbeeld is

`getFlexChart(pie,opening_verdelingomgzakennaarsoort)` waarbij de eerste parameter *pie* aangeeft dat het gewenste diagram een taartvorm is en waarbij *opening_verdelingomgzakennaarsoort* de naam van de query is (tbquery.dvcode) die een jsonstring met de nodige informatie genereert. (OpenWave levert deze queries niet standaard uit: U dient deze zelf te definiëren)

query

De char.js library verwacht een Json string met daarin metadata (zoals labels en kleuren) en data (aantallen). Bij een pie (taart-diagram) ziet zo'n JSON-string er (naar bovenstaand voorbeeld) zo uit (lijkt eigenlijk er veel op staafdiagram, maar nu moeten we per label een aparte kleur definiëren). Vanwege de beperking in de breedte van de tegel zijn in bovenstaand voorbeeld de labelnamen afgekort.

```
{
  "labels": ["Verzk", "Tzcht", "Klcht", "Mldng", "Cncpt", "Overg"],
  "datasets": [{
    "backgroundColor": ["rgba(0, 255, 255, 1)", "rgba(0, 77, 153, 1)",
"rgba(254, 164, 0, 1)", "rgba(100, 100, 100, 1)", "rgba(200, 200, 200,
1)", "rgba(300, 300, 300, 1)"],
    "borderColor": "rgba(255, 0, 0, 1)",
    "borderWidth": 0.5,
    "data": [904,56,9,334,33,431]
  }]
}
```

De uitkomst van de query moet dus bovenstaande string opleveren waarbij de de aantallen zaken (de data) opgehaald moeten worden uit tbomgvergunning verzameld op

- Verzk (verzoeken): (tbsoortomgverg.dvsoortproc = R of U)
- Tzcht (toezicht): (tbsoortomgverg.dvsoortproc = C of T of H)
- Klcht (klacht): (tbsoortomgverg.dvsoortproc = K)
- Mldng (melding): (tbsoortomgverg.dvsoortproc = M)
- Cncpt (conceptverzoek/vooroverleg): (tbsoortomgverg.dvsoortproc = V)
- Overig: (tbsoortomgverg.dvsoortproc is ongelijk aan bovenstaande)

```
select '{
  "labels": ["Verzk", "Tzcht", "Klcht", "Mldng", "Cncpt", "Overg"],
  "datasets": [{
    "backgroundColor": ["rgba(0, 255, 255, 1)", "rgba(0, 77, 153, 1)",
"rgba(254, 164, 0, 1)", "rgba(100, 100, 100, 1)", "rgba(200, 200, 200,
1)", "rgba(300, 300, 300, 1)"],
    "borderColor": "rgba(255, 0, 0, 1)",
    "borderWidth": 0.5,
    "data": [' ||
      (select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and b.dvsoortproc in ('R','U')) || ','
||
      (select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and b.dvsoortproc in ('C','T','H')) ||
',' ||
    ]
  }]
}
```

```
(select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and b.dvsoortproc = 'K') || ',' ||
(select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and b.dvsoortproc = 'M') || ',' ||
(select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and b.dvsoortproc = 'V') || ',' ||
(select count(*) from tbomgvergunning a inner join
tbsoortomgverg b on (a.dnkeysoortomgverg = b.dnkey) where a.ddbesluitdatum
is null and a.ddvernietigd is null and
coalesce(instr('RUCTHKMV',b.dvsoortproc),0) = 0) || ']'
}]
}'
```

In bovenstaande voorbeelden zijn de diagrammen toegevoegd op bestaande tegels van het openingsportaal. dat kan, maar kost natuurlijk ook tijd. Een alternatief is een apart portal te maken met tegels die diagrammen laten zien.

From:
<https://doc.open-wave.nl/> - Documentatie

Permanent link:
https://doc.open-wave.nl/doku.php/openwave/1.30/applicatiebeheer/instellen_inrichten/grafieken?rev=1716277382

Last update: 2024/05/21 09:43

