Dokumentation - Kortkomponent

Version 2.0.0

Indholdsfortegnelse

1. Indledning
    1. Kom i gang
        1. Seneste version
        2. Versioneret version
        3. Versionering
2. API-nøgle
3. Map
    1. Metoder
4. Profiler
    1. Standard profil
5. Typer
    1. Feature
        1. Markør
        2. Bygning
    2. MatrikelKey
    3. Geometrier
        1. Geometri
        2. Punkt
        3. Polygon
6. Kontroller
    1. Kontroltyper
7. Lag
    1. Lag typer
8. Eksempler
    1. Minimalt
    2. Med ikoner

Indledning

Dette dokument udgør en overordnet teknisk beskrivelse omkring opsætningen af kortkomponenten udviklet af JO Informatik. Dokumentet giver et overblik over mulige parametre samt eksempler på anvendelse.

Kortkomponenten benytter sig af OpenLayers v. 10.7.0 (https://www.openlayers.org). Herefter benævnt som *OL*.

Kom i gang

For at benytte kortbiblioteket skal der blot inkluderes to filer på sin html side. Filerne udbydes en seneste udgave, der kan ændre sig over tid og en versioneret udgave, der ikke bliver ændret.

Seneste version

Javascript:

- https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.js

Css:

- https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.css

```
<link rel="stylesheet" href="https://basiskort.bbr.dk/lib/bbr.basemap-
2.0.0.min.css">
<script src="https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.js">
</script>
```

Kortkomponenten initialiseres på følgende måde (Klasserne skal prefixes med UFST):

```
<script>
  new UFST.Map('divID');
</script>
```

Versioneret version

Inkluderes på html side på samme måde som seneste udgave, dog skal der benyttes følgende url:

Javascript:

- https://basiskort.bbr.dk/lib/bbr.basemap-[version].min.js

Css:

- https://basiskort.bbr.dk/lib/bbr.basemap-[version].min.css

Hvor [version] er angivet ved x.x.x. Eksempel bbr.basemap-2.0.0.min.css

```
<link rel="stylesheet" href="https://basiskort.bbr.dk/lib/bbr.basemap-
2.0.0.min.css">
<script src="https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.js">
</script>
```

Komplet liste over versioner kan findes her

I version 1 kunne man benytte følgende links for at altid køre med seneste version. Det videreføres ikke. i version 2 skal man altid angive en specifik version.

```
<link rel="stylesheet"
href="https://basiskort.bbr.dk/lib/bbr.basemap.min.css">
<script src="https://basiskort.bbr.dk/lib/bbr.basemap.min.js"></script>
```

API-nøgle

Kortkomponenten henter alle data fra Datafordeleren, og der følger **ikke** en API-nøgle med komponenten. Du skal derfor angive din egen abonnementsnøgle til Datafordeleren med `UFST.BaseMapDAF.setApiKey()` **inden** kortet oprettes:

```
UFST.BaseMapDAF.setApiKey("min-api-noegle");

var map = new UFST.Map("mapDiv");
```

Nøglen benyttes til alle kald til Datafordeleren (WMS, WMTS, WFS og GraphQL).

Oprettes et kort uden en nøgle, skrives en fejl i browserens konsol (én gang pr. side), og alle kald til Datafordeleren vil fejle. Kortet kaster ikke en exception - det renderes blot uden data.

Metoder

- `UFST.BaseMapDAF.setApiKey(apiKey)` Sætter nøglen. Kaldes den med en tom værdi, skrives en fejl i konsollen, og den hidtidige nøgle bevares
- `UFST.BaseMapDAF.hasApiKey()` Returnerer `true`, hvis der er sat en nøgle

Map

Opretter et kort vha. kortkomponenten. Kortet forventer at koordinater m.m. er angivet i UTM32 format.

Syntax

```
var map = new UFST.Map(mapDiv, [options]);
```

Parametre

- `mapDiv {string}` Id på den divider der skal bruges til kortet
- `options {MapOptions}` En række valgfri parametre til kortet

MapOptions

- `view {ViewOptions}` Angiver hvor kortet startes. Hvis ikke angivet starte kortet på et zoom niveau over hele landet
- `controls {(string|Control)[]}` Et array med kontroller, der skal tilføjes kortet link
- `layers {(string|Layer)[]}` Et array med lag, der skal tilføjes kortet link
- `interactions {InteractionOptions}` Styrer hvilke interaktioner der er i kortet link
- `events {EventOptions}` Angiver hvilke map events, der skal tilknyttes kortet

- `showGeoLocationButton {boolean}`: Angiver om geolokationsknappen skal vises på kortet. `True`/`False`. Hvis ikke angivet, så er den `false`
- `activateDefaultShortcut {boolean}` Aktiverer standardgenveje for de tilføjede kontroller. Hvis ikke angivet, så er den `false` - Valgfrit
- `controlContainerClass {string}` CSS-klasse for containeren med kontroller. Hvis ikke angivet, bruges `ctrl-container` - Valgfrit

ViewOptions

- `startupCenter {[number, number]}` Koordinaten hvor kortet skal centreres omkring
- `startupZoom {number}` Zoom niveauet der startes på - heltal, fra 0 og opefter
- `startupExtent {[number, number, number, number]}` En boundingbox der skal startes op på
- `maxZoom {number}` Det højeste zoomniveau, der anvendes ved zoom til geometri - Valgfrit
- `grunddataKeys {MatrikelKey[]}` Nøgler der kan bruges til at starte kortet op på bestemte matrikler link
- `grunddataKeysOptions {GrunddataKeysOptions}` Indstillinger for opslag af grunddatanøgler - Valgfrit
- `markers {Marker[]}` Markører i kortet bl.a. bygninger og tekniske anlæg link
- `profile {string}` En profil der afgør hvordan kortet ser ud og hvad der er standard
- `useCluster {boolean}` Angiver om markører grupperes. Hvis ikke angivet, så er den `true` - Valgfrit
- `padding {number[]}` Indvendig afstand ved zoom til et område - Valgfrit
- `defaultDuration {number}` Varighed i millisekunder for zoom-animationer. Hvis ikke angivet, så er den `300` - Valgfrit
- `skipAnimateOnFirstLoad {boolean}` Slår animation fra ved den første zoomning af kortet. Hvis ikke angivet, så er den `false` - Valgfrit

GrunddataKeysOptions

- `allowInvalidGrunddataKeys {boolean}` Tillader et resultat, selv om enkelte grunddatanøgler er ugyldige. Hvis ikke angivet, så er den `true` - Valgfrit
- `allowExternalCallToFailed {boolean}` Tillader eksterne opslag efter et mislykket opslag. Hvis ikke angivet, så er den `true` - Valgfrit
- `allowParserError {boolean}` Tillader et resultat ved parsefejl. Hvis ikke angivet, så er den `true` - Valgfrit
- `timeout {number}` Timeout i millisekunder for opslag. Hvis ikke angivet, bruges komponentens standardtimeout - Valgfrit

EventOptions

- `onMapLoaded {() => void}` Callback der kaldes, når kortet er indlæst
- `onMapError {(ErrorCodes, object) => void}` Callback der kaldes, hvis et opslag af grunddatanøgler fejler
- `onClick {[number, number], MapBrowserEvent}` Callback der bliver kaldt, når der klikkes på kortet og der ikke klikkes på en markør. Ved dobbelt klik bliver denn kaldt to gange. Hvis der ikke er konfigureret nogle af den andre onMarker*Clicked event, så giver den klik event over hele kortet

- `onMarkerClicked {Marker[], MapBrowserEvent}` Callback bliver kaldt når der klikkes på en markør. `MapBrowserEvent`* er OLs lowlevel mouse klik object
- `onMarkerUnClicked {Marker[], MapBrowserEvent}` Callback bliver kaldt når den valgte markør bliver fravalgt. Bemærk den kaldes kun hvis der klikkes på en ny marker. `MapBrowserEvent`* er OLs lowlevel mouse klik object
- `onMarkerHovered {Marker[], MapBrowserEvent}` Callback bliver kaldt når mousen er over en markør (bemærk når flere markører er clustered, bliver denne ikke kaldt). `MapBrowserEvent`* er OLs lowlevel mouse klik object
- `onMarkerUnHovered {Marker[], MapBrowserEvent}` Callback bliver kaldt når mousen fjernes fra en markør (bemærk når flere markører er clustered, bliver denne ikke kaldt). `MapBrowserEvent`* er OLs lowlevel mouse klik object
- `onMapViewChanged {(centerCoordinate, mapSize, zoomLevel) => boolean}` Callback der kaldes, når brugeren er færdig med at flytte eller zoome kortet

* `MapBrowserEvent` kan være null, hvis kaldet ikke er aktiveret af et museklik.

Returnerer `{Map}` Kort instans link

Eksempel

Eksempel på argumenter:

```
options: {
    view: {
      startupCenter: [,],
      startupZoom: 2,
      startupExtent: [,,,],
      grunddataKeys: [...],
      markers: [...],
      profile: 'retbbr'
    },
    controls: [...],
    layers: [...],
    interactions: {...},
}
```

Eksempel på kort:

```
var map = new UFST.Map('container',
    {
        view: {
            startupCenter: [600000, 6237500]
        },
        layers: ['ortofoto']
    }
);
```

Metoder

Returnere en instans af objektet, der kan benyttes til videre modificering. Instansen har følgende metoder:

`addControl(control)`

Tilføjer en kontrol til kortet

Parametre

- `control {Control}` Den kontrol der skal tilføjes

Returnerer `{void}`

Eksempel

```
map.addControl(new UFST.SimpleLayerSwitcherControl(null));
```

`getControls()`

Returnere alle kontroller der er tilføjet kortet

Returnerer `{Control[]}` en liste af kontroller

Eksempel

```
const count = map.getControls().length;
```

`getControlById(id)`

Finder en kontrol ud fra dens id.

Parametre

- `id {string}` Id'et på kontrollen

Returnerer `{Control | null}` Kontrollen, eller `null` hvis den ikke findes

`addLayer(layer)`

Tilføjer et lag til kortet

Parametre

- `layer {Layer}` Det lag, der skal tilføjes

Returnerer `{void}`

Eksempel

```
map.addLayer(
    new UFST.WMTSOrtofoto(
        new UFST.WMSOptions({
            visible: true
        })
));
```

`getLayers()`

Returnere alle lag der er tilføjet kortet

Returnerer `{Layer[]}` en liste af lag

Eksempel

```
const count = map.getLayers().length;
```

`getLayerByName(name)`

Finder et lag ud fra dets navn.

Parametre

- `name {string}` Lagets navn

Returnerer `{Layer | null}` Laget, eller `null` hvis det ikke findes

`setZoomLevel(zoomLevel)`

Sætter zoom niveauet

Parametre

- `zoomLevel {number}` Det zoom niveau der skal sættes, fra 0 og opefter

Returnerer `{void}`

Eksempel

```
map.setZoomLevel(10);
```

`setMaxZoomLevel()`

Zoomer så langt ind i kortet som er muligt

Returnerer `{void}`

Eksempel

```
map.setMaxZoomLevel();
```

`getMaxZoomLevel()`

Få kortets maksimalt mulige zoom niveau

Returnerer `{number}` Det højest tilladte zoom niveau

Eksempel

```
const maxPossibleZoomLevel = map.getMaxZoomLevel();
```

`getZoomLevel()`

Få kortets aktuelle zoom niveau

Returnerer `{number}` Det aktuelle zoom niveau

Eksempel

```
const zoomLevel = map.getZoomLevel();
```

`setCenter(x, y)`

Sætter kortets center koordinat

Parametre

- `x {number}` X koordinaten
- `y {number}` Y koordinaten

Returnerer `{void}`

Eksempel

```
map.setCenter(600000, 6237500);
```

`setPadding(padding)`

Sætter den indvendige afstand, der bruges ved zoom til et område.

Parametre

- `padding {number[]}` Afstand i rækkefølgen top, højre, bund, venstre

Returnerer `{void}`

`zoomToGeometry(geom)`

Zoomer ind i kortet så hele geometrien kan ses

Parametre

- `geom {Geometry | number[] | number[][][]}` En geometri (point, polygon)

Returnerer `{void}`

Eksempel

```
map.zoomToGeometry(new UFST.Polygon([[600000, 6237500], [600000, 6240500],
[610000, 6240500], [610000, 6237500], [600000, 6237500]]));
map.zoomToGeometry([[[600000, 6237500], [600000, 6240500], [610000,
6240500], [610000, 6237500], [600000, 6237500]]]);
map.zoomToGeometry([600000, 6237500]);
```

`zoomToGeometries(geoms)`

Zoomer ind i kortet så alle geometrier kan ses

Parametre

- `geoms { Geometry[] | number[][] | number[][][][]}` Geometrier (point, polygon)

Returnerer `{void}`

Eksempel

```
map.zoomToGeometries([new UFST.Polygon([[600000, 6237500], [600000,
6240500], [610000, 6240500], [610000, 6237500], [600000, 6237500]])]);
map.zoomToGeometries([[[[600000, 6237500], [600000, 6240500], [610000,
6240500], [610000, 6237500], [600000, 6237500]]]]);
map.zoomToGeometries([[600000, 6237500], [600100, 6240500]]);
```

`zoomToExtent(extent)`

Zoomer ind i kortet til en bestemt boundingbox

Parametre

- `extent {[number, number, number, number]}` En boundingbox

Returnerer `{void}`

Eksempel

```
map.zoomToExtent([724729.735, 6215479.714, 724778.228, 6215551.81]);
```

goToExtent(extent)

Flytter kortet til en boundingbox uden at zoome længere ind end det aktuelle zoomniveau.

Parametre

- `extent {[number, number, number, number]}` En boundingbox

Returnerer `{void}`

fit(extent, [options])

Tilpasser kortets visning til en boundingbox.

Parametre

- `extent {[number, number, number, number]}` En boundingbox
- `options {FitOptions}` Indstillinger for tilpasningen - Valgfrit

FitOptions

- `size {[number, number]}` Kortets størrelse i pixels - Valgfrit
- `duration {number}` Animationens varighed i millisekunder - Valgfrit
- `maxZoom {number}` Højeste zoomniveau - Valgfrit
- `minResolution {number}` Laveste opløsning - Valgfrit
- `padding {[number, number, number, number]}` Indvendig afstand i rækkefølgen top, højre, bund, venstre - Valgfrit

Returnerer `{void}`

zoomToGeoLocation()

Zoomer til brugerens geo-lokation.

Returnerer `{void}`

setMaskAndZoomToGeometry(geom, [zoom])

Sætter en maske og zoomer ind i kortet så hele geometrien kan ses

Parametre

- `geom {Geometry}` En geometri (point, polygon)
- `zoom {boolean}` Hvis true, så zoomes der til de nye geometrier. Hvis false, så zoomes der ikke. Hvis den ikke angives så er den true - Valgfrit

Returnerer `{void}`

Eksempel

```
map.setMaskAndZoomToGeometry(new UFST.Polygon([[600000, 6237500], [600000, 6240500], [610000, 6240500], [610000, 6237500], [600000, 6237500]]));
```

`setMaskAndZoomToGeometries(geoms, [zoom])`

Sætter en maske og zoomer ind i kortet så alle geometrier kan ses

Parametre

- `geoms {Geometry[]}` Geometrier (point, polygon)
- `zoom {boolean}` Hvis true, så zoomes der til de nye geometrier. Hvis false, så zoomes der ikke. Hvis den ikke angives så er den true - Valgfrit

Returnerer `{void}`

Eksempel

```
map.setMaskAndZoomToGeometries([new UFST.Polygon([[600000, 6237500], [600000, 6240500], [610000, 6240500], [610000, 6237500], [600000, 6237500]])]);
```

`updateMaskGeometry(geom)`

Sætter en maske ud fra en geometri

Parametre

- `geom {Geometry}` En geometri (point, polygon)

Returnerer `{void}`

Eksempel

```
map.updateMaskGeometry(new UFST.Polygon([[600000, 6237500], [600000, 6240500], [610000, 6240500], [610000, 6237500], [600000, 6237500]]));
```

`updateMaskGeometries(geoms)`

Sætter en maske ud fra en liste af geometrier

Parametre

- `geoms {Geometry[]}` Geometrier (point, polygon)

Returnerer `{void}`

Eksempel

```
map.updateMaskGeometries([new UFST.Polygon([[600000, 6237500], [600000, 6240500], [610000, 6240500], [610000, 6237500], [600000, 6237500]])]);
```

`getMaskGeometry()`

Få den aktuelle maske

Returnerer `{Geometry[] | null}` De aktuelle geometrier, eller `null` hvis der ikke er sat en maske

Eksempel

```
const hasMask = (map.getMaskGeometry() || []).length > 0;
```

`zoomToMask()`

Zoomer ind, så den aktuelle maske kan ses.

Returnerer `{void}`

`clearMask()`

Fjerner den aktuelle maske.

Returnerer `{void}`

`clear()`

Fjerner markører, maske og gemte grunddatanøgler.

Returnerer `{void}`

`getCurrentExtent()`

Få den aktuelt synlige boundingbox

Returnerer `{[number, number, number, number]}`

Eksempel

```
const bbox = map.getCurrentExtent();
```

`isCoordinateInView(x, y)`

Tjekker om et koordinat er i den synlige del af kortet

Parametre

- `x {number}` X koordinaten
- `y {number}` Y koordinaten

Returnerer `{boolean}` en værdi der indikere om koordinatet er synligt i kortet

Eksempel

```
const isInMap = map.isCoordinateInView(600000, 6237500);
```

`setMatrikel(landsejerlavskode, matrikelNr, [zoom])`

Sætter maske der passer til matriklen

Parametre

- `landsejerlavskode {number}` landsejerlavskode
- `matrikelNr {string}` matrikelNr
- `zoom {boolean}` Hvis true, så zoomes der til de nye grunddata keys. Hvis false, så zoomes der ikke. Hvis den ikke angives så er den true - Valgfrit

Returnerer `{Promise<void>}`

Eksempel

```
    map.setMatrikel(2000652, '85b');
```

`setGrunddataKey(key, [append], [zoom])`

Sætter maske der passer til grunddata nøglen

Parametre

- `key {(MatrikelKey)}` Grunddata nøgle link
- `append {boolean}` Hvis true, så tilføjes de nye nøgler til de eksisterende nøgler. Hvis false, så nulstilles grunddata nøglerne. Hvis den ikke angives så er den false - Valgfrit
- `zoom {boolean}` Hvis true, så zoomes der til de nye grunddata keys. Hvis false, så zoomes der ikke. Hvis den ikke angives så er den true - Valgfrit

Returnerer `{Promise<void>}`

Eksempel

```
map.setGrunddataKey({ landsejerlavskode: 2000652, matrikelNr: '85b' });
```

setGrunddataKeys(keys, [append], [zoom])

Sætter maske der passer til alle grunddata nøgler

Parametre

- `key {(MatrikelKey)[]}` Grunddata nøgler link
- `append {boolean}` Hvis true, så tilføjes de nye nøgler til de eksisterende nøgler. Hvis false, så nulstilles grunddata nøglerne. Hvis den ikke angives så er den false - Valgfrit
- `zoom {boolean}` Hvis true, så zoomes der til de nye grunddata keys. Hvis false, så zoomes der ikke. Hvis den ikke angives så er den true - Valgfrit

Returnerer `{Promise<void>}`

Eksempel

```
  map.setGrunddataKeys([{ landsejerlavskode: 2000652, matrikelNr: '85b' },{
landsejerlavskode: 2000652, matrikelNr: '84l' }]);
```

getGrunddataKeys()

Henter de grunddatanøgler, der senest blev fundet ved opslag.

Returnerer `{GrunddataKeyResult[]}` En liste over de fundne grunddatanøgler

clearGrunddataKeys()

Fjerner de gemte grunddatanøgler. Metoden fjerner ikke kortets maske.

Returnerer `{void}`

getMarkers()

Henter kortets markører

Returnerer `marker {Marker}` Markør link

Eksempel

```
var marker = map.getMarkers();
```

setMarker(marker)

Sætter markør/ikon i kortet

Parametre

- `marker {Marker}` Markør link

Returnerer `{void}`

Eksempel

```
  map.setMarker({ id: 1, x: 724609.68, y: 6215471.97, shortname: '1', icon:
'erhverv', color: 'sikker' });
```

`setMarkers(markers)`

Sætter markører/ikoner i kortet

Parametre

- `markers {Marker[]}` Et array af markører link

Returnerer `{void}`

Eksempel

```
  map.setMarkers([{ id: 1, x: 724609.68, y: 6215471.97, shortname: '1',
icon: 'erhverv', color: 'sikker' },
                        { id: 2, x: 724603.71, y: 6215441.29, shortname:
'2', icon: 'smaabygning', color: 'sikker' }]);
```

`zoomToMarkers()`

Zoomer ind, så alle markører kan ses.

Returnerer `{void}`

`zoomToMarker(id)`

Zoomer ind på en markør.

Parametre

- `id {string}` Markørens id

Returnerer `{void}`

`goToMarker(id)`

Flytter kortet til en markør uden at zoome længere ind end det aktuelle zoomniveau.

Parametre

- `id {string}` Markørens id

Returnerer `{void}`

`getMarkerById(id)`

Finder en markør ud fra dens id.

Parametre

- `id {string}` Markørens id

Returnerer `{Marker | null}` Markøren, eller `null` hvis den ikke findes

`removeMarker(marker)`

Fjerner en markør fra kortet.

Parametre

- `marker {Marker}` Markøren der skal fjernes

Returnerer `{void}`

`clearMarkers()`

Fjerner alle markører fra kortet.

Returnerer `{void}`

`setMarkerActiveState(id, active, [deactivateOthers])`

Sætter en markørs aktive tilstand.

Parametre

- `id {string}` Markørens id
- `active {boolean}` Den aktive tilstand
- `deactivateOthers {boolean}` Deaktiverer alle andre markører. Hvis ikke angivet, så er den `true` - Valgfrit

Returnerer `{void}`

`setMarkerSelectedState(id, selected, [deactivateOthers])`

Sætter en markørs valgte tilstand.

Parametre

- `id {string}` Markørens id
- `selected {boolean}` Den valgte tilstand

- `deactivateOthers {boolean}` Fravælger alle andre markører. Hvis ikke angivet, så er den `true` - Valgfrit

Returnerer `{void}`

`setMarkerCoordinate(id, coordinate)`

Opdaterer en markørs koordinat.

Parametre

- `id {string}` Markørens id
- `coordinate {[number, number]}` Markørens nye koordinat

Returnerer `{void}`

`setMarkerProperty(id, property, value)`

Opdaterer en egenskab på en markør.

Parametre

- `id {string}` Markørens id
- `property {MarkerProperty}` Egenskaben der skal opdateres
- `value {any}` Den nye værdi

Returnerer `{void}`

`updateMapSize()`

Opdaterer kortets størrelse efter en ændring af kortets container.

Returnerer `{void}`

`setControlContainerClass(className)`

Opdaterer CSS-klassen på containeren med kortets kontroller.

Parametre

- `className {string}` Den nye CSS-klasse

Returnerer `{void}`

Profiler

Kortet kan startes op i forskellige profiler, hvor udseenet på ikonerne m.m. kan være forskelligt.

Standard profil

Hvis der ikke angives en profil, så benyttes standard profilen og den har følgende ikoner:

- `erhverv` Ikonet for erhvers bygninger

- `spoergsmaal` Ikonet for et spørgsmålstegn
- `advarsel` Ikonet for advarsel
- `ukendt-bygning` Ikonet for en ukendt bygning
- `smaabygning` Ikonet for småbyggeri
- `fritidsformaal` Ikonet for fritidsformål
- `carport-garage` Ikonet for garage/carport
- `olietank` Ikonet for olietank
- `silo` Ikonet for silo
- `halmfyr-biogas-mv` Ikonet for et halmfyr/giogas/...
- `vandanlaeg` Ikonet for et vandanlæg
- `vindmoelle` Ikonet for en vindmølle
- `energi` Ikonet for et energi anlæg
- `ensilage-planlager` Ikonet for ensilage/planlager
- `tank` Ikonet for en tank
- `andet-anlaeg` Ikonet for andre anlæg
- `ukendt-TEK` Ikonet for et ukendt teknisk anlæg
- `industri` Ikonet for industri
- `helaarsbeboelse` Ikonet for helårsbeboelse
- `landbrug` Ikonet for landbrug
- `institution` Ikonet for institution
- `idraetsformaal` Ikonet for idrætsformål

Typer

Feature

Angiver base klassen for en feature, har følgende egenskaber:

- `id {string}` Id'et på featuren
- `x {number}` X-koordinaten i UTM 32 koordinatsystem. For typer der har en geometri, der har en udbredelse, er denne null (f.eks. har en polygon som geometri)
- `y {number}` Y-koordinaten i UTM 32 koordinatsystem. For typer der har en geometri, der har en udbredelse, er denne null (f.eks. har en polygon som geometri)

Markør (Marker - inherits Feature)

Angiver en markør, har følgende egenskaber:

- `titel {string}` Titlen på markøren
- `shortname {string}`: Et kort navn til markøren,
- `icon {string}` Markørens ikon fra predefineret liste, se listen af mulige ikoner under link
- `color {string}` Farven på markøren. Enten en predefineret farve (sikker, usikker, standard og signatur) eller en css farvekode

Bygning (Bygning - inherits Feature)

Angiver en bygning, har følgende egenskaber:

MatrikelKey

Angiver en matrikel, har følgende egenskaber:

- `landsejerlavskode {number}` Landsejerlavskode
- `matrikelNr {string}` Matrikelnr

Geometrier

Representere de forskellige geometrier, der kan benyttes.

Geometri (Geometry)

Base klasse for geometrier

Punkt (Point - inherits Geometry)

Syntax

```
var punkt = new UFST.Point(coordinat);
```

Parametre

- `coordinat {[number, number]}` X og y koordinaten for punktet

Returnerer `{Point}` Instans af punktet

Eksempel

```
var punkt = new UFST.Point([600000, 6237500]);
```

Polygon (Polygon - inherits Geometry)

Syntax

```
var polygon = new UFST.Polygon(coordinates);
```

Parametre

- `coordinates {number[][]}` En liste af x og y koordinater for polygonen på formen [[x1, y1], [x2, y2], ..., [x1, y1]]

Returnerer `{Polygon}` Instans af polygonen

Eksempel

```
var polygon = new UFST.Polygon([[600000, 6237500], [600000, 6240500],
```

```
  [610000, 6240500], [610000, 6237500], [600000, 6237500]]);
```

Kontroller

I kortkomponenten findes der følgende kontroller:

- `attribution` Viser copyright for de enkelte lag
- `default` Standard kontroller
- `geolocation` Kontrol til hentning af brugerens geolokation
- `matrikelselector` Giver brugeren mulighed for at vælge en matrikel på kortet
- `scaleline` Målestok
- `simplelayerswitcher` Skift baggrundslag
- `zoom` Zoom ind/ud

Standard aktive kontroller (hvis intet andet er angivet):

- `attribution`
- `scaleline`
- `simplelayerswitcher`
- `zoom`

Kontrol typer

Kontrol (Control)

Opretter en base kontrol - Nedarv fra denne kontrol, når der skal oprettes en ny kontrol i kortkomponenten.

Syntax

```
var control = new UFST.Control([options]);
```

Parametre

- `options {ControlOptions}` En række parametre til kontrollen

ControlOptions

- `id {string}` Et id for kontrollen - Påkrævet

Returnerer `{Control}` En instance af kontrollen

`show()`

Viser kontrollen på kortet.

Parametre `{void}`

Returnerer `{void}`

`hide()`

Skjuler kontrollen på kortet.

Parametre `{void}`

Returnerer `{void}`

Geolokations (GeoLocation - inherits Control)

Udbyder en kontrol til at navigere til brugerens geolokation.

Syntax

```
var control = new UFST.GeoLocationControl([options]);
```

Parametre

- `options {GeoLocationControlOptions}` En række valgfri parametre til geolokations kontrollen, samt dem der er angivet i `{ControlOptions}`

GeoLocationControlOptions

- `showButton {boolean}` Vis geolokationsknappen på kortet. Hvis det ikke angives, så er værdien `false` - Valgfrit

Returnerer `{GeoLocationControl}` En instance af geolokation kontrollen

`zoomToGeoLocation()`

Zoomer til brugeren geolocation.

Parametre `{void}`

Returnerer `{void}`

Matrikelvælger (MatrikelSelectorControl - inherits Control)

Kontrollen giver brugeren mulighed for at kikke på et punkt i kortet og så kan kortkomponenten slå op via koordinaten hvilke veje/kommuner punktet ligger på Ved opslaget hentes der data fra følgende kilder:

```
* `Datafordeleren WFS` finder det gældende jordstykke ved det valgte
koordinat
```

- `Datafordeleren GraphQL` henter jordstykke, lodflade, BFE og ejerlav
- `Datafordeleren GraphQL` henter ejendommens beliggenhedsadresse og kommune

Bemærk for at få at vide hvilken matrikel, der er klikket på, skal metoden `OnShowProperty` angives i options. Se beskrivelse under `MatrikelSelectorControlOptions`.

Syntax

```
var control = new UFST.MatrikelSelectorControl([options]);
```

Parametre

- `options {MatrikelSelectorControlOptions}` En række valgfri parametre til signaturforklaringen, samt dem der er angivet i `{ControlOptions}`

MatrikelSelectorControlOptions

- `IsNewProperty {MatrikelSelectorResponse} => boolean` En metode, der kaldes for at tjekke om brugeren har klikket på en ny ejendom. Hvis metoden returnere true, så vises popupen med resultatet fra opslaget. Hvis den retuurnere false, så vises ingen popup. Hvis den ikke angives, så returneres der altid true
- `OnShowProperty {MatrikelSelectorResponse} => void` Metoden bliver kald når brugeren klikker på knappen `Vis ejendom`. Hermed kan der skiftet til at vise en ny ejendom på kortet. Hvis denne metode ikke erklæres, så sker der ikke noget når brugeren klikker på `Vis ejendom` knappen
- `MaxZoomLevel {int}` Max zoom niveau for hvornår, der skal sendes kald afsted til Datafordeleren. Som standard er denne sat til zoom niveau 11

Returnerer `{MatrikelSelectorControl}` En instance af matrikel selector kontrollen

MatrikelSelectorResponse

Objektet indeholder følgende egenskaber:

- `Geometri` Ejendommens geometri inkl type, crs og koordinater
- `BfeNr` Ejendommens BFEnr
- `MatrikelNr` Ejendommens matrikelnr
- `EjerlavNavn` Ejendommens landsejerlavnavn
- `EjerlavKode` Ejendommens landsejerlavskode
- `JordstykkeId` Matriklen jordstykkeid
- `AdresseBetegnelse` Adressebetegnelsen til ejendommen
- `VejNavn` Vejnavnet på ejendommen
- `HusNummerTekst` Hus nummeret på ejendommen
- `PostNr` Postnr for ejendommen
- `PostDistrikt` Postdistrikt for ejendommen
- `KommuneNavn` Navnet på kommunen som ejendommen ligger i
- `EtageBetegnelse` Evt etage betegnelse
- `DørBetegnelse` Evt dør betegnelse
- `DAWARaw` Udfaset kompatibilitetsfelt. Feltet er altid `null` i version 2.0

Eksempel

```
let options = UFST.MatrikelSelectorControlOptions({
    OnShowProperty: (property) => {
        // Skift til ny ejendom på baggrund af indholdet af property
objektet
    }
});
```

Simpel lagvælger (OverviewMapLayerSwitcherControl - inherits Control)

Danner en simpel lag vælger, der skifter mellem en række lag - Som standard mellem ortofoto og skærmkort.

Syntax

```
var control = new UFST.OverviewMapLayerSwitcherControl([options]);
```

Parametre

- `options {OverviewMapLayerSwitcherControlOptions}` En række valgfri parametre til lagvælgeren, samt dem der er angivet i `{ControlOptions}`

OverviewMapLayerSwitcherControlOptions

- `toogleLayers {Layer[]}` En række lag der toogle imellem. Hvis ikke angivet toogles mellem ortofoto og skærmkort - Valgfrit

Returnerer `{LegendControl}` En instance af bbr signaturforklaringen

Signaturforklaring (LegendControl - inherits Control)

Danner base klasse for en signatur forklaring. Denne komponent indeholder logikken til at vise en divider med åbn/luk funktionalitet. Nedarv fra denne klasse for at lave en ny signaturforklaring.

Syntax

```
var control = new UFST.LegendControl([options]);
```

Parametre

- `options {LegendControlOptions}` En række valgfri parametre til signaturforklaringen, samt dem der er angivet i `{ControlOptions}`

LegendControlOptions

Returnerer `{LegendControl}` En instance af bbr signaturforklaringen

BBR-Signaturforklaring (BBRLegendControl - inherits LegendControl)

Danner en BBR signaturforklaring, hvor der vises signaturer for den makører der er vist på kortet.

Syntax

```
var control = new UFST.BBRLegendControl([options]);
```

Parametre

- `options {BBRLegendControlOptions}` En række valgfri parametre til signaturforklaringen, samt dem der er angivet i `{LegendControlOptions}`

BBRLegendControlOptions

- `showMarkerIcons {boolean}` Viser ikoner for de viste markører i kortet. Hvis det ikke angives, så er værdien `true` - Valgfrit
- `showStatusIcons {boolean}` Viser forskellige status ikoner (som f.eks. Advarsels ikonet). Hvis det ikke angives, så er værdien `true` - Valgfrit

Returnerer `{BBRLegendControl}` En instance af bbr signaturforklaringen

Lag

I kortkomponenten findes der følgende lag:

- `ortofoto` Ortofoto* - Se `WMS Ortofoto`
- `orto` Ortofoto (WMS/WMTS)* - Se `Group – Ortofoto`
- `skaermkort` Skærmkort* - Se `WMS Skærmkort`
- `skaerm` Skærmkort (WMS/WMTS)* - Se `Group – Skærmkort`
- `matrikel` Matrikel lag
- `adresse` Adresse lag
- `vejnavn` Vejnavn lag
- `mask` Et lag der kan bruges til at fokusere på bestemte områder (hvidt klæde)
- `bygning` Et lag der viser bygningspolygoner indefor bestemte områder
- `default` standard lag

*baggrundslag

Standard aktive lag (hvis intet andet er angivet):

- `orto`
- `skaerm`
- `matrikel`
- `adresse`
- `vejnavn`
- `mask`

Rækkefølgen lagene angives i har betydning for hvordan lagene tegnes. Først i listen tegnes først i kortet med de andre lag ovenpå.

Lag typer

Layer

Opretter et base kortlag - Nedarv fra dette lag, når der skal oprettes et nyt lag i kortkomponenten.

Syntax

```
var layer = new UFST.Layer(layer, [options]);
```

Parametre

- `layer {object}` Kortlaget der skal tilføjet kortkomponenten
- `options {LayerOptions}` En række valgfri parametre til laget

LayerOptions

- `id {string}` Angiver id'et på laget. Dette skal være unik blandt alle lag. Hvis det ikke angives, så dannes et tilfædigt - Valgfrit
- `visible {boolean}` Skal laget vises som standard. Hvis det ikke angives så er den true - Valgfrit
- `opacity {number}` Angiver lagets opacitet - Mellem 0 og 1. Hvis det ikke angives så er den 1.0 - Valgfrit
- `minResolution {number}` Hvilken nedre resolution skal laget vises ved. Ved null eller undefined, så er der ingen nedre grænse. Hvis det ikke angives så er den null - Valgfrit
- `maxResolution {number}` Hvilken øvre resolution skal laget vises ved. Ved null eller undefined, så er der ingen øvre grænse. Hvis det ikke angives så er den null - Valgfrit
- `type {string}` Angiver typen på laget, bruges når der tjekkes hvilken lagtype instansen er dannet fra. Hvis det ikke angives så er den null - Valgfrit
- `isBackgroundLayer {boolean}` Er laget et baggrundslag - true/false. Hvis det ikke angives så er den false - Valgfrit
- `events {LayerEventType}` Events der kan tilføje laget. Hvis denne ikke sættes er der ingen events - Valgfrit
- `isWMTS {boolean}` Er laget et WMTS lag - true/false. Hvis det ikke angives så er den false - Valgfrit
- `name {string}` Angiver navnet på laget. Hvis det ikke angives så er den null - Valgfrit

Returnerer `{Layer}` Layer lag instans

WMS (WMSLayer - inherits Layer)

Opretter et WMS lag.

Syntax

```
var layer = new UFST.WMSLayer(layer, [options]);
```

Parametre

- `layer {object}` Kortlaget der skal tilføjes kortkomponenten
- `options {WMSLayerOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{LayerOptions}`

WMSLayerOptions

Returnerer `{WMSLayer}` WMS lag instans

Gruppelag (GroupLayer - inherits Layer)

Opretter et lag basseret på en samling af lag

Syntax

```
var map = new UFST.GroupLayer(layer, [options]);
```

Parametre

- `layer {object[]}` En række kortlag der skal tilføjet kortkomponenten
- `options {LayerOptions}` En række valgfri parametre til laget - Se under layer

Returnerer `{GroupLayer}` Gruppe lag instans

Vector (VectorLayer - inherits Layer)

Opretter et vector lag

Syntax

```
var layer = new UFST.VectorLayer([options]);
```

Parametre

- `options {VectorLayerOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{LayerOptions}`

VectorLayerOptions

- `style {object}` Stilen som laget skal vises via. Ved null benyttes standard visning. Hvis det ikke angives så er den `null` - Valgfrit
- `styles {object[]}` En liste af stile som laget skal vises via. Ved tom eller null benyttes standard visning. Hvis det ikke angives så er den `[]` - Valgfrit

- `source {object}` Definere vector lagets source. Ved null oprettes en ny. Hvis det ikke angives så er den `null` - Valgfrit
- `updateWhileAnimating {boolean}` Angiver om laget skal opdateres under animering - `true/false`. Hvis det ikke angives så er den `false` - Valgfrit
- `updateWhileInteracting {boolean}` Angiver om laget skal opdateres under interaktion fra brugeren - `true/false`. Hvis det ikke angives så er den `false` - Valgfrit

Bemærk:

- Der i VectorOptions kun skal sættes enten `style` eller `styles` og ikke dem begge.
- Hvis type er `null`, så sættes den til `vector.Base`

Returnerer `{VectorLayer}` Returnere en instans af laget, der kan benyttes til videre modificering af vector laget.

Instansen har følgende metoder:

`addFeature(feature)`

Tilføjer en feature til vektor laget.

Parametre

- `feature {Feature}` Den feature, der skal tilføjes

Returnerer `{void}`

`addFeatures(features[])`

Tilføjer en liste af features til vektor laget.

Parametre

- `features {Feature[]}` En liste af features, der skal tilføjes

Returnerer `{void}`

`getFeatures()`

Returnerer de features der findes på laget

Parametre

- `{void}`

Returnerer `{Feature[]}` En liste af features, der findes på laget

`getFeatureById(id)`

Returnerer en feature via dens id

Parametre

- `id {string}` Id'et på den features, der skal hentes

Returnerer `{Feature}` Den feature der matcher id'et eller `null`, hvis den ikke findes

`clearFeatures()`

Fjerner alle features fra laget.

Parametre

- `{void}`

Returnerer `{void}`

`removeFeature(feature)`

Fjerner en bestem feature fra laget.

Parametre

- `feature {Feature}` Fjerner en specifik feature fra laget

Returnerer `{void}`

`hasFeatures()`

Tester om laget indeholder features.

Parametre

- `{void}`

Returnerer `{boolean}` `true` hvis laget har en eller flere features ellers `false`

WMTS (WMTSLayer - inherits Layer)

Opretter et WMTS lag - bruges til nedarvning fra andre WMTS lag

Syntax

```
var layer = new UFST.WMTSLayer(wmtsOptions, [options]);
```

Parametre

- `wmtsOptions {WMTSLayerOptions | WMTSLayerDAFOptions}` Indstillinger for WMTS laget
- `options {LayerOptions}` En række valgfri parametre til laget - Se under layer

Brug `WMTSLayerOptions` for generelle kald til WMTS kilder f.eks. `Kortforsyningen/Dataforsyningen` eler brug `WMTSLayerDAFOptions` hvis der skal hentes fra `Datafordeleren`.

WMTSLayerOptions

- `matrixIds {string[]}` Angiver WMTS matrixId'er - Hvis dette ikke angives benyttes `["L00", "L01", "L02", "L03", "L04", "L05", "L06", "L07", "L08", "L09", "L10", "L11", "L12", "L13"]`
- `resolutions {number[]}` Angiver WMTS resolutions - Hvis dette ikke angives benyttes `[1638.4, 819.2, 409.6, 204.8, 102.4, 51.2, 25.6, 12.8, 6.4, 3.2, 1.6, .8, .4, .2]`
- `orgin {number[]}` Angiver WMTS laget orgin (x, y) - Hvis dette ikke angives benyttes `120000, 6500000`
- `matrixSet {string}` Angiver WMTS laget matrixSet - Hvis dette ikke angives benyttes `View1`

WMTSLayerDAFOptions

- `matrixIds {string[]}` Angiver WMTS matrixId'er - Hvis dette ikke angives benyttes `["L00", "L01", "L02", "L03", "L04", "L05", "L06", "L07", "L08", "L09", "L10", "L11", "L12", "L13"]`
- `resolutions {number[]}` Angiver WMTS resolutions - Hvis dette ikke angives benyttes `[1638.4, 819.2, 409.6, 204.8, 102.4, 51.2, 25.6, 12.8, 6.4, 3.2, 1.6, .8, .4, .2]`

Returnerer `{WMTSLayer}` WMTS lag instans

Adresse (WMSAdresseLayer - inherits WMSLayer)

Opretter et lag, der viser adresser på kortet. Data hentes fra Datafordeler og laget `adgangsadresser`. Laget vises først ved resolution `6.4`. Kan overskrives via options parameteren.

Syntax

```
var layer = new UFST.WMSAdresseLayer([options]);
```

Parametre

- `options {WMSOptions}` En række valgfri parametre til laget

Returnerer `{WMSAdresseLayer}` Adresse lag instans

Matrikel (WMSMatrikelLayer - inherits WMSLayer)

Opretter et lag, der viser matrikler på kortet. Data hentes fra Datafordelerens service `MATRIKLEN2/MatGaeldendeOgForeloebigWMS` og laget `Jordstykke_Gaeldende`.

Syntax

```
var layer = new UFST.WMSMatrikelLayer([options]);
```

Parametre

- `options {WMSOptions}` En række valgfri parametre til laget

Returnerer `{WMSMatrikelLayer}` Matrikel lag instans

WMS - Ortofoto (WMSOrtofotoLayer - inherits WMSLayer)

Opretter et Ortofoto WMS lag. Data hentes fra Datafordelerens service `orto_foraar` og laget `orto_foraar`.

Syntax

```
var layer = new UFST.WMSOrtofotoLayer([options]);
```

Parametre

- `options {WMSOptions}` En række valgfri parametre til laget

Returnerer `{WMSOrtofotoLayer}` Ortofoto lag instans

WMS - Skærmkort (WMSSkaermkortLayer - inherits WMSLayer)

Opretter et Skærmkort WMS lag. Data hentes fra Datafordelerens service `topo_skaermkort` og laget `topo_skaermkort`.

Syntax

```
var layer = new UFST.WMSSkaermkortLayer([options]);
```

Parametre

- `options {WMSSkaermkortLayerOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{WMSOptions}`

WMSSkaermkortLayerOptions

- `useDimLayer {boolean}` Skal skærmkortet vises i dæmpet udgave eller normal udgave. Ved dæmpet udgave benyttes lagnavnet `dtk_skaermkort_daempet`. Hvis dette ikke angives, så er den `true` - Valgfrit
- `useGrayLayer {boolean}` Skal skærmkortet vises i grå udgave. Benyttes kun når `useDimLayer` er `false`. Hvis dette ikke angives, så er den `false` - Valgfrit

Returnerer `{WMSSkaermkortLayer}` Skærmkort lag instans

Vejnavn (WMSVejnavnLayer - inherits WMSLayer)

Opretter et vejnavne WMS lag. Data hentes fra Dataforsyningens service `forvaltning` og laget `navne_basis_ortofoto`. Laget vises først ved resolution `6.4`.

Syntax

```
var layer = new UFST.WMSVejnavnLayer([options]);
```

Parametre

- `options {WMSOptions}` En række valgfri parametre til laget

Returnerer `{WMSVejnavnLayer}` Vejnavne lag instans

WMTS - Ortofoto (WMTSOrtofotoLayer - inherits WMTSLayer)

Opretter et ortofoto WMTS lag. Data hentes fra Datafordelerens service `GeoDanmarkOrto` og laget `orto_foraar_wmts`.

Syntax

```
var map = new UFST.WMTSOrtofotoLayer([options],[wmtsOptions]);
```

Parametre

- `options {LayerOptions}` En række valgfri parametre til laget
- `wmtsOptions {WMTSOptions}` En række valgfri wmts parametre til laget

Returnerer `{WMTSOrtofotoLayer}` Ortofoto wmts lag instans

WMTS - Skærmkort (WMTSSkaermkortLayer - inherits WMTSLayer)

Opretter et skærmkort WMTS lag. Data hentes fra Datafordelerens service `Dkskaermkort`.

Syntax

```
var layer = new UFST.WMTSSkaermkortLayer([options],[wmtsOptions]);
```

Parametre

- `options {WMTSSkaermkortLayerOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{LayerOptions}`
- `wmtsOptions {WMTSOptions}` En række valgfri wmts parametre til laget

WMTSSkaermkortLayerOptions

- `useDimLayer {boolean}` Viser skærmkortet i dæmpet tilstand. Ved `true` hentes data fra `topo_skaermkort_daempet`. Hvis den ikke er angivet, så er den `true` - Valgfrit

- `useGrayLayer {boolean}` Viser skærmkortet i grå tilstand. Benyttes kun når `useDimLayer` er `false`. Hvis den ikke er angivet, så er den `false` - Valgfrit

Returnerer `{WMTSSkaermkortLayer}` Skærmkort wmts lag instans

Mask (VectorMaskLayer - inherits VectorLayer)

Danner et maske lag på kortet.

Syntax

```
var layer = new UFST.VectorMaskLayer([options]);
```

Parametre

- `options {VectorMaskOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{VectorOptions}`

VectorMaskOptions

- `maskGeometry {Geometry[]}` En liste med de geomtrier, der skal tegnes maske udfra - Valgfrit
- `showUnionedGeometries {boolean}` Samler polygonerne til hele geometrier. Hvis ikke angivet er denne `true`- Valfrit
- `maskLimitResolution {number}` Angiver resolution på hvornår det hvide klæde forsvinder og grunden kun vises med en blå outline. Hvis ikke angivet er den `1.4` - Valfrit

Returnerer `{VectorMaskLayer}` Returnerer en instans af laget, der kan benyttes til videre modificering.

Instansen har følgende metoder:

`updateMaskGeometry(geometry)`

Opdaterer lagets maske

Parametre

- `geometry {Geometry[]}` De geometrier som masken skal bestå af

Returnerer `{void}`

`supportMask()`

Angiver om laget skal opdateres hvis kortets maske opdateres. Altid true

Parametre

- `{void}`

Returnerer `{boolean}`

Bygning (VectorBygningLayer - inherits Vector)

Bygningslaget viser de bygningspolygoner, der ligger indenfor et given område. Som standard bliver de tegnet med hvide linier. Datakilden for laget er Datafordeleren.

Syntax

```
var layer = new UFST.VectorBygningLayer([options]);
```

Parametre

- `options {VectorBygningOptions}` En række valgfri parametre til laget, samt dem der er angivet i `{VectorOptions}`

VectorBygningOptions

- `events {VectorBygningLayerEventOptions}` En liste med events for laget. Nedarver fra `LayerEventType`
- `includeOnlyBygningWithMatchingMarkerId {boolean}` Hvis true, så der kun vises polygoner for bygninger, der har et matchene id fra markør listen. Bemærk dette id skal matche det id, der findes i BBRUUID (case er vigtig). Hvis en bygning har et id og den ikke matcher, så vises bygningen ikke. Hvis den ikke har et id, så prøves næste check (`includeOnlyBygningWithinGeometry`). Hvis denne ikke sættes er den true - Valgfrit
- `includeOnlyBygningWithinGeometry {boolean}` Hvis true så vises polygoner for bygninger, hvis polygonen ligger indenfor en af viste grunde. Hvis denne ikke sættes er den true - Valgfrit

VectorBygningLayerEventOptions

- `onError {(ErrorCodes, object) => void}` En callback, der bliver kaldt hvis der sker en fejl under hentningen af data
- `onSuccess {(Bygning[]) => void}` En callback, der bliver kaldt efter data er hentet, med de features (Bygninger), der er fundet

Returnerer `{VectorBygningLayer}` Returnerer en instans af laget, der kan benyttes til videre modificering.

Instansen har følgende metoder:

`updateMaskGeometry(geometry)`

Opdaterer lagets maske

Parametre

- `geometry {Geometry[]}` De geometrier som masken skal bestå af

Returnerer `{void}`

`supportMask()`

Angiver om laget skal opdateres hvis kortets maske opdateres. Altid true

Parametre

- `{void}`

Returnerer `{boolean}`

`retryLoad()`

Forsøger at kalde lagets eksterne kilde igen for at hente data.

Parametre

- `{void}`

Returnerer `{void}`

Eksempel

```
let bygningsLag = new UFST.VectorBygningLayer(
    new UFST.VectorBygningLayerOptions(
        {
            events: {
                onError: function (errorCode, e) {
                    ...
              }
            }
            ...
        }
    )
);
```

Group - Ortofoto (GroupOrtofotoLayer - inherits Group)

Opretter et lag med både ortofoto i WMTS og WMS udgave. Laget skifter automatisk, når der ikke er flere zoom niveauer i WMTS laget. Er en gruppe af lagene `WMS - Ortofoto` og `WMTS - Ortofoto`.

Syntax

```
var layer = new UFST.GroupOrtofotoLayer([wmsLayerOptions],
[wmtsLayerOptions], [groupLayerOptions], [wmtsOptions]);
```

Parametre

- `wmsLayerOptions {WMSLayerOptions}` En række valgfri parametre til wms laget
- `wmtsLayerOptions {LayerOptions}` En række valgfri parametre til wmts laget
- `groupLayerOptions {LayerOptions}` En række valgfri parametre til gruppe laget
- `wmtsOptions {WMTSLayerOptions}` En række WMTS options parametre til wmts laget

Returnerer `{GroupOrtofotoLayer}` Ortofoto gruppe lag instans

Group - Skærmkort (GroupSkaermkortLayer - inherits Group)

Opretter et lag med både skærmkort i WMTS og WMS udgave. Laget skifter automatisk, når der ikke er flere zoom niveauer i WMTS laget. Er en gruppe af lagene `WMS - Skærmkort` og `WMTS - Skærmkort`.

Syntax

```
var layer = new UFST.GroupSkaermkortLayer([wmsLayerOptions],
[wmtsLayerOptions], [groupLayerOptions], [wmtsOptions]);
```

Parametre

- `wmsLayerOptions {WMSLayerOptions}` En række valgfri parametre til wms laget
- `wmtsLayerOptions {LayerOptions}` En række valgfri parametre til wmts laget
- `groupLayerOptions {LayerOptions}` En række valgfri parametre til gruppe laget
- `wmtsOptions {WMTSLayerOptions}` En række WMTS options parametre til wmts laget

Returnerer `{GroupSkaermkortLayer}` Skærmkort gruppe lag instans

Interactions

I kortkomponenten findes der følgende interaktioner:

InteractionOptions

- `useCtrlOnMouseWheelZoom (boolean)` Man kan kun zoome med mussehjul når ctrl holdes nede
- `enableMouseWheelScroll (boolean)` Angiver om det er muligt at zoome via. mussehjul
- `enableMapMouseDrag (boolean)` Angiver om det er muligt at trække i kortet for navigering
- `enablePinchRotate (boolean)` Angiver om det er muligt at rotere kortet ved brug af fingrene
- `enablePinchZoom (boolean)` Angiver om det er muligt at zoome i kortet ved brug af fingrene
- `enableDoubleClickZoom (boolean)` Angiver om det er muligt at zoome ved at dobbeltklikke
- `shiftDragZoom (boolean)` Angiver om det er muligt at zoome til et område ved at trække med Shift-tasten holdt nede

Følgende standard bruges hvis intet andet er angivet:

```
interactions: {
    useCtrlOnMouseWheelZoom: false,
    enableMouseWheelScroll: true,
    enableMapMouseDrag: true,
    enablePinchRotate: true,
    enablePinchZoom: true,
    enableDoubleClickZoom: true,
```

```
  shiftDragZoom: true,
}
```

Eksempler

Forskellige brugs eksempler kan findes her

Minimalt

Minimalt eksempel, der viser kortet ved standard zoom niveau og center punkt.

```
<!DOCTYPE html>
<html>
    <head>
        <title>BaseMap eksempel</title>
        <meta charset="utf-8" />
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-
scale=1">
        <link rel="stylesheet"
href="https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.css" />
        <script src="https://basiskort.bbr.dk/lib/bbr.basemap-
2.0.0.min.js"></script>
        <script type="text/javascript">
            function onload() {
                UFST.BaseMapDAF.setApiKey("min-api-noegle");

                var map = new UFST.Map("mapDiv");
            }
        </script>
    </head>
    <body onload="onload()">
        <h1>Basiskort - Eksempel</h1>
        <div id="mapDiv" style="border:1px solid
black;width:800px;height:600px;"></div>
    </body>
</html>
```

Med ikoner

Viser kortet zoomet ind på en matrikel med markører.

```
<!DOCTYPE html>
<html>
    <head>
        <title>BaseMap eksempel</title>
        <meta charset="utf-8" />
```

```
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-
scale=1">
        <link rel="stylesheet"
href="https://basiskort.bbr.dk/lib/bbr.basemap-2.0.0.min.css" />
        <script src="https://basiskort.bbr.dk/lib/bbr.basemap-
2.0.0.min.js"></script>
        <script type="text/javascript">
            function onload() {
                UFST.BaseMapDAF.setApiKey("min-api-noegle");

                var map = new UFST.Map("mapDiv", {
                view: {
                    grunddataKeys: [
                        { landsejerlavskode: 2000652, matrikelNr: '85b' },
                    ],
                    markers: [
                        { id: 1, x: 724609.68, y: 6215471.97, shortname:
'1', icon: 'erhverv', color: 'sikker' },
                        { id: 2, x: 724603.71, y: 6215441.29, shortname:
'2', icon: 'smaabygning', color: 'sikker' },
                        { id: 3, x: 724591.37, y: 6215463.01, shortname:
'T1', warning: true, icon: 'olietank', color: 'usikker'},
                        { id: 4, x: 724594.37, y: 6215470.81, shortname:
'T2', warning: true, icon: 'olietank', color: 'usikker' },
                        { id: 5, x: 724621.82, y: 6215474.52, shortname:
'T3', warning: true, icon: 'olietank', color: 'usikker' },
                        { id: 6, x: 724623.12, y: 6215477.52, shortname:
'T4', warning: true, icon: 'olietank', color: 'usikker' },
                    ]
                }
            });
            }
        </script>
    </head>
    <body onload="onload()">
        <h1>Basiskort - Eksempel</h1>
        <div id="mapDiv" style="border:1px solid
black;width:800px;height:600px;"></div>
    </body>
</html>
```