
DEATH STRANDING

Generic Sensor API

Koa

Koa

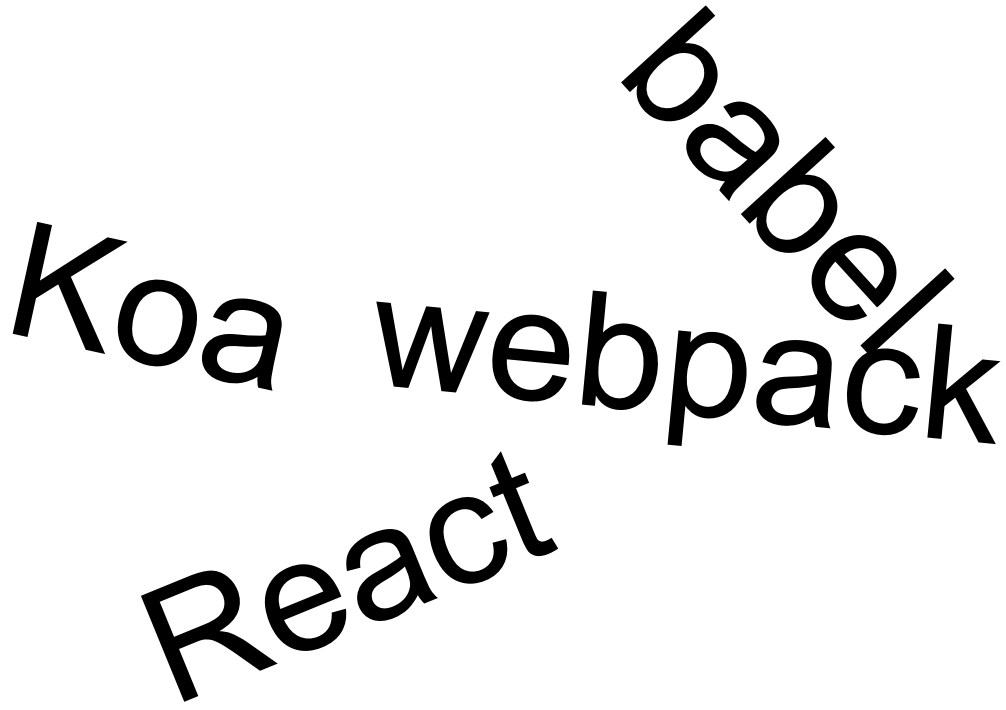
Backbone

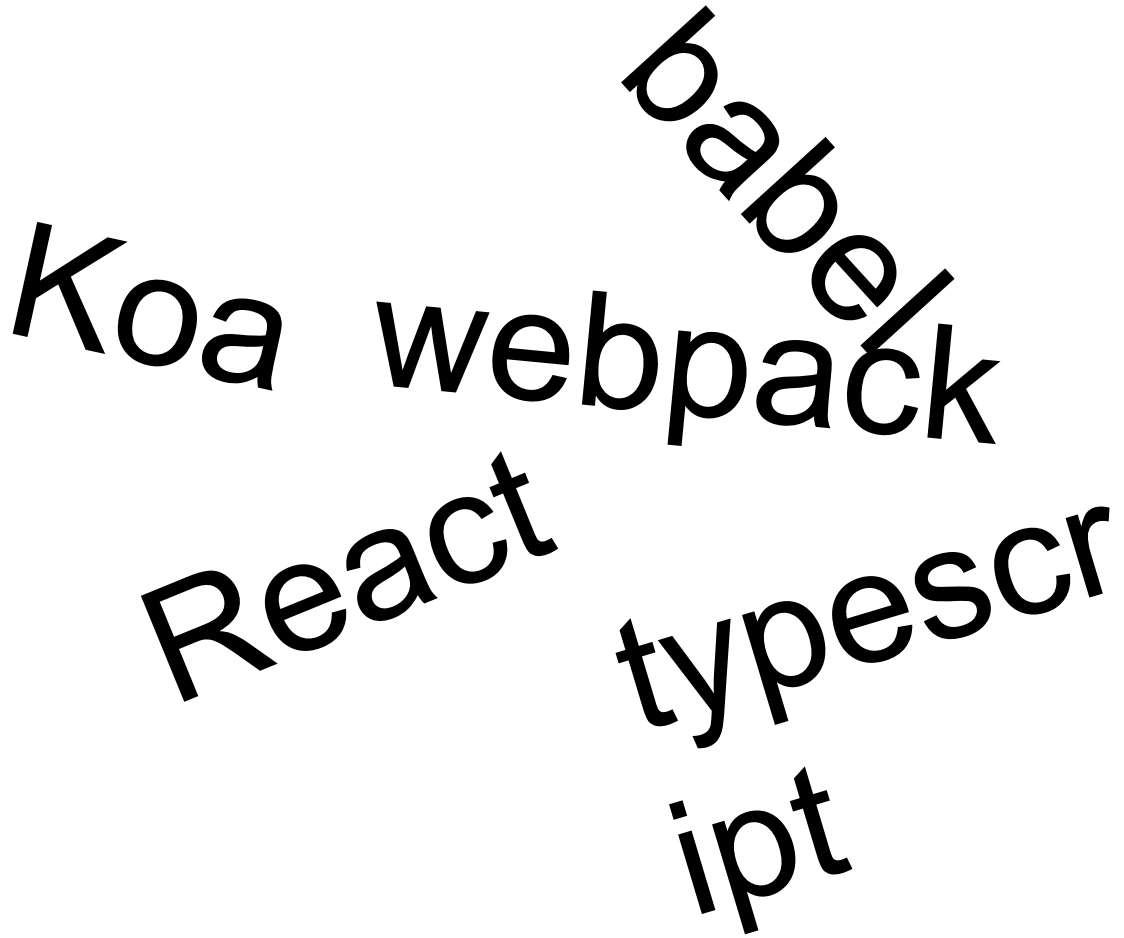
Koa

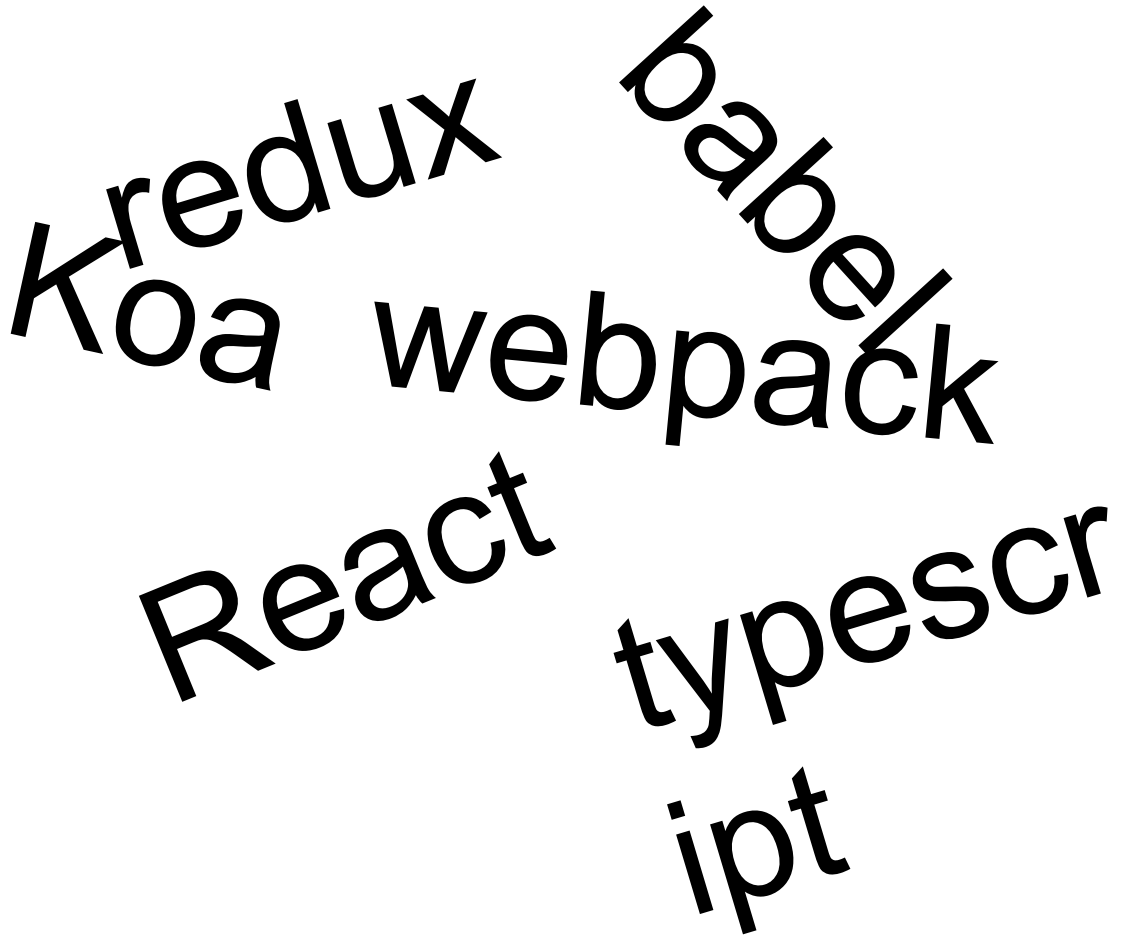
React

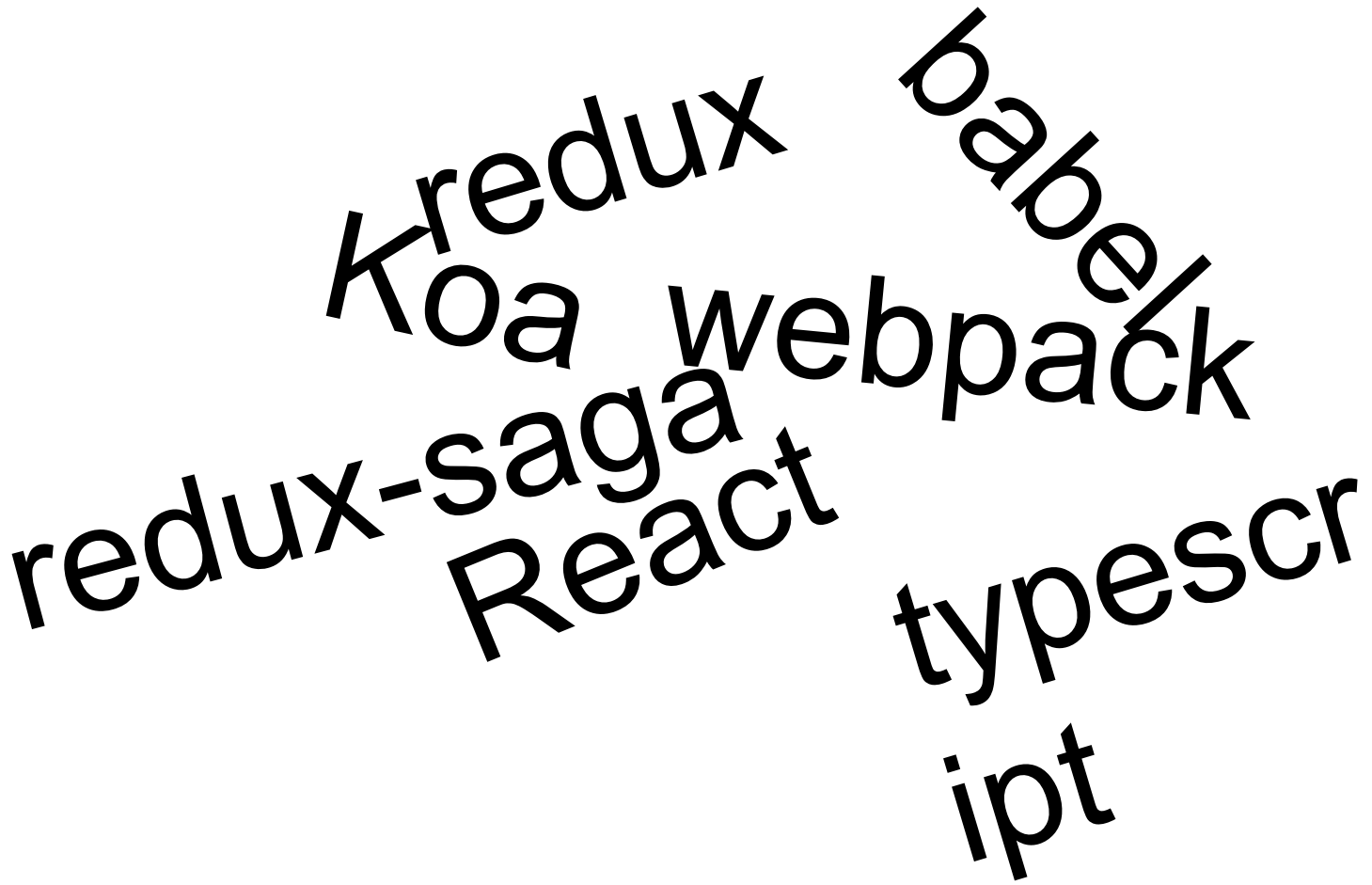
Koa webpack

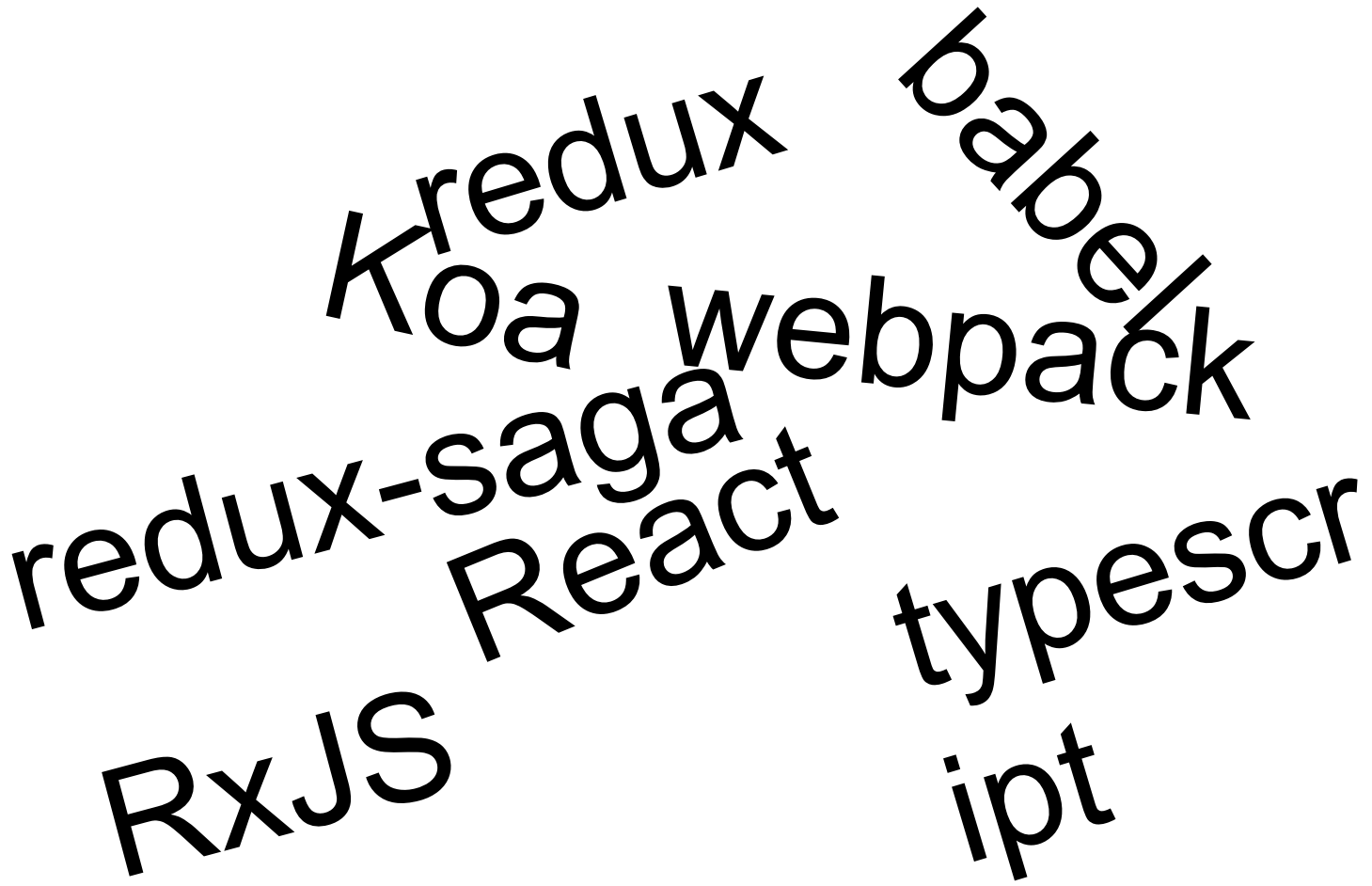
React

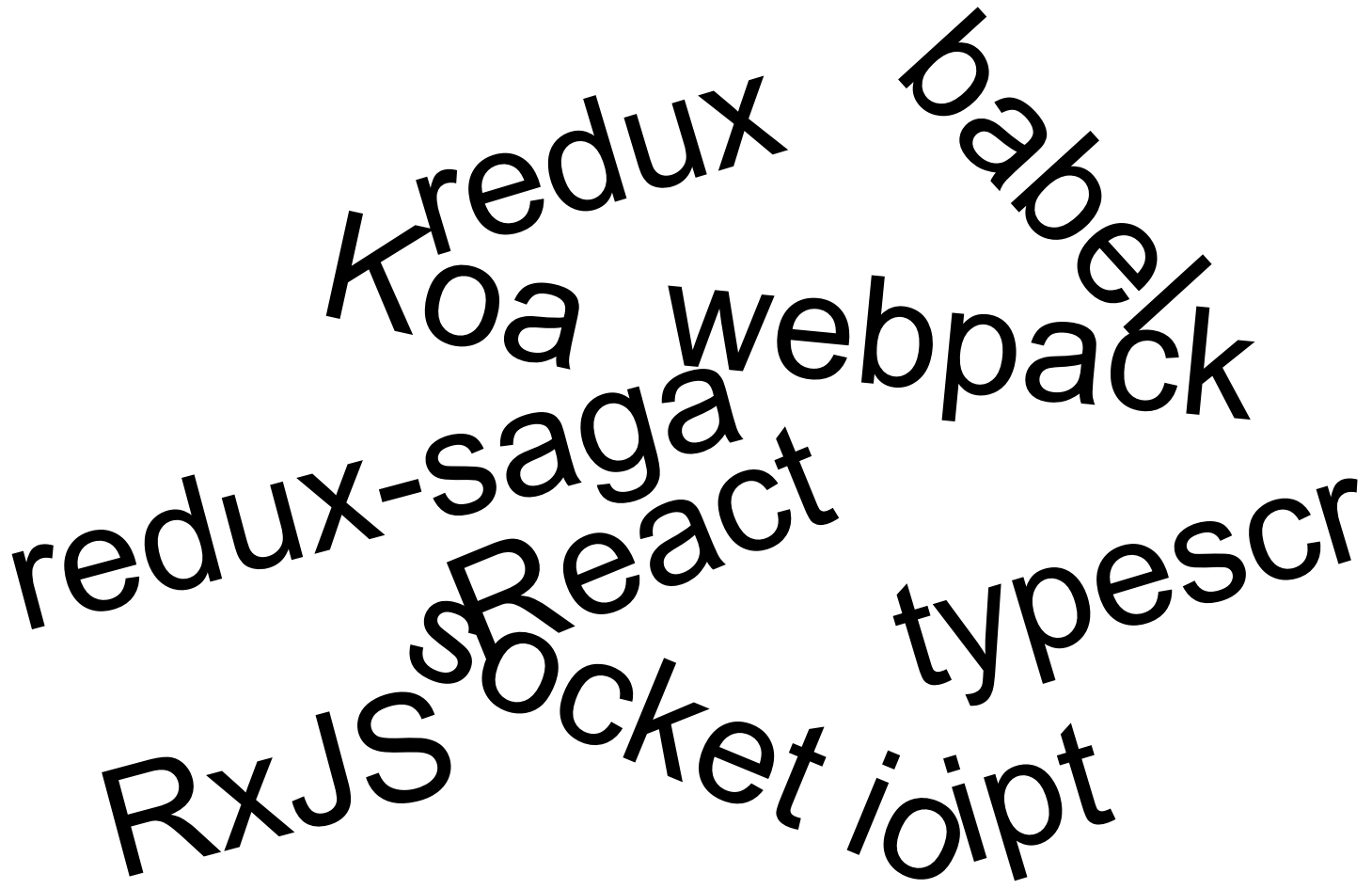

babel
Koa
webpack
React


babel
Koa
webpack
React
typescr
ipt


redux
babel
Koa
webpack
React
typescr
ipt


redux
babel
Koa
webpack
redux-saga
React
typescr
ipt


redux
babel
Koa
webpack
redux-saga
React
typescr
RxJS
ipt


redux
babel
Koa
webpack
redux-saga
React
typescr
RxJS
socket.io
ipt


redux
babel
Wasted
redux-
React
typescr
RxJS
ocket
io
ipt

Generic Sensor API

AbsoluteOrientationSensor

Generic Sensor API

AbsoluteOrientationSensor

RelativeOrientationSensor

Generic Sensor API

AbsoluteOrientationSensor

RelativeOrientationSensor

Accelerometer

Generic Sensor API

AbsoluteOrientationSensor

RelativeOrientationSensor

Accelerometer

LinearAccelerationSensor

Generic Sensor API

AbsoluteOrientationSensor

RelativeOrientationSensor

Accelerometer

LinearAccelerationSensor

Gyroscope

Generic Sensor API

AbsoluteOrientationSensor

RelativeOrientationSensor

Accelerometer

LinearAccelerationSensor

Gyroscope

AmbientLightSensor

Generic Sensor API

AbsoluteOrientationSensor

Accelerometer

Gyroscope

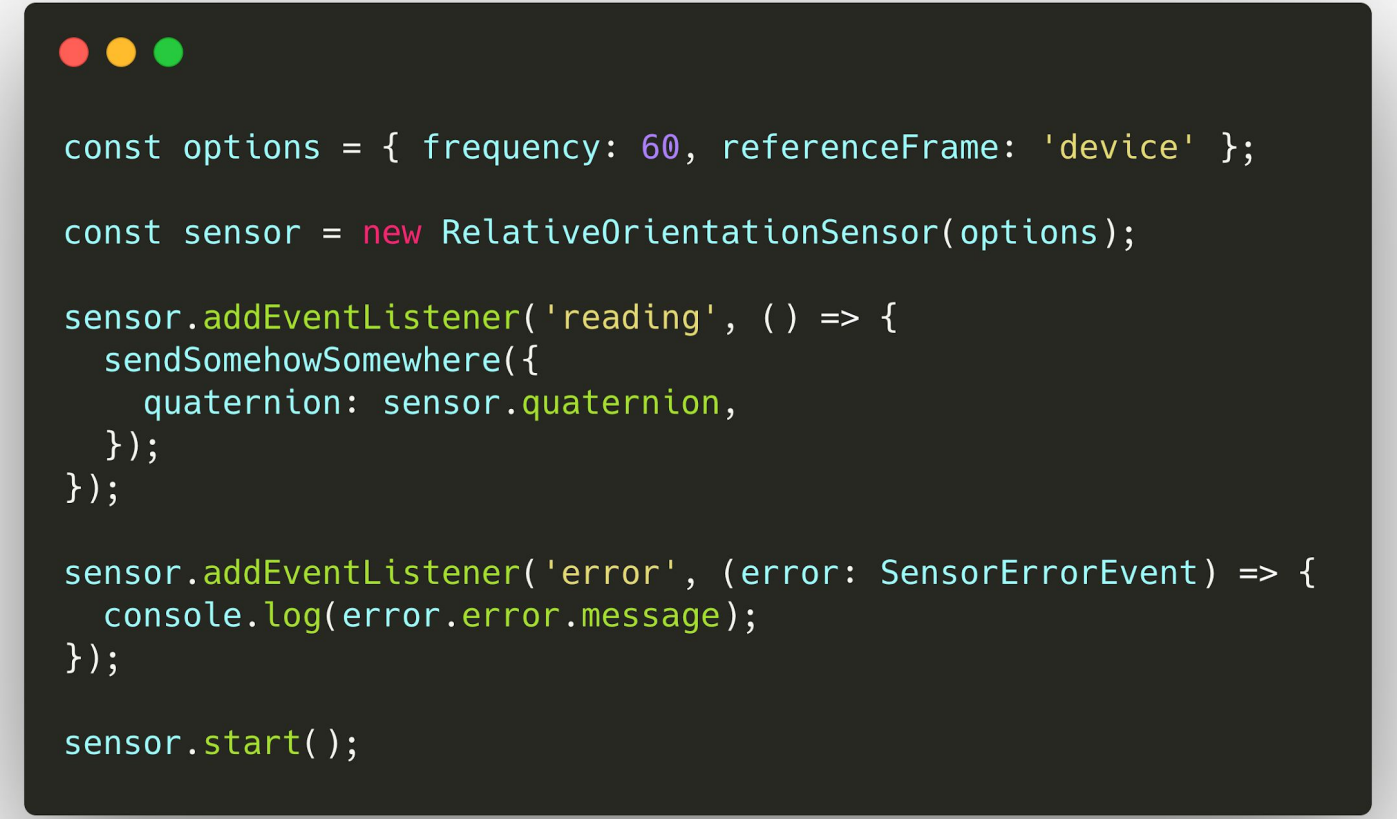
Magnetometer

RelativeOrientationSensor

LinearAccelerationSensor

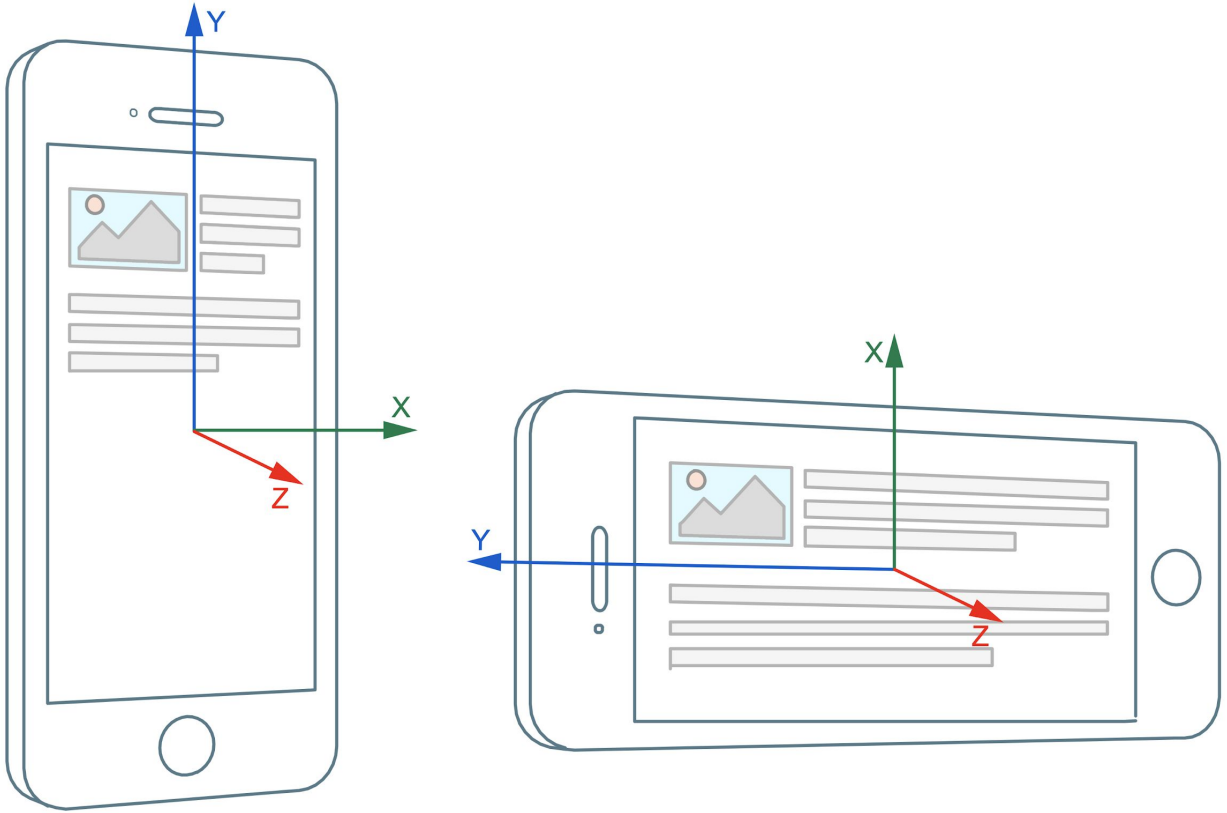
AmbientLightSensor

```
Promise.all([
      navigator.permissions.query({ name: 'accelerometer' }),
      navigator.permissions.query({ name: 'magnetometer' }),
      navigator.permissions.query({ name: 'gyroscope' }),
      navigator.permissions.query({ name: 'ambient-light-sensor' }),
      navigator.permissions.query({ name: 'geolocation' }),
]).then(results => {
  if (result.state == 'granted') {
    // Действуем
  } else if (result.state == 'prompt') {
    // Запрашиваем
  }
}).catch((error) => {
  // Что-то пошло не так
});
```

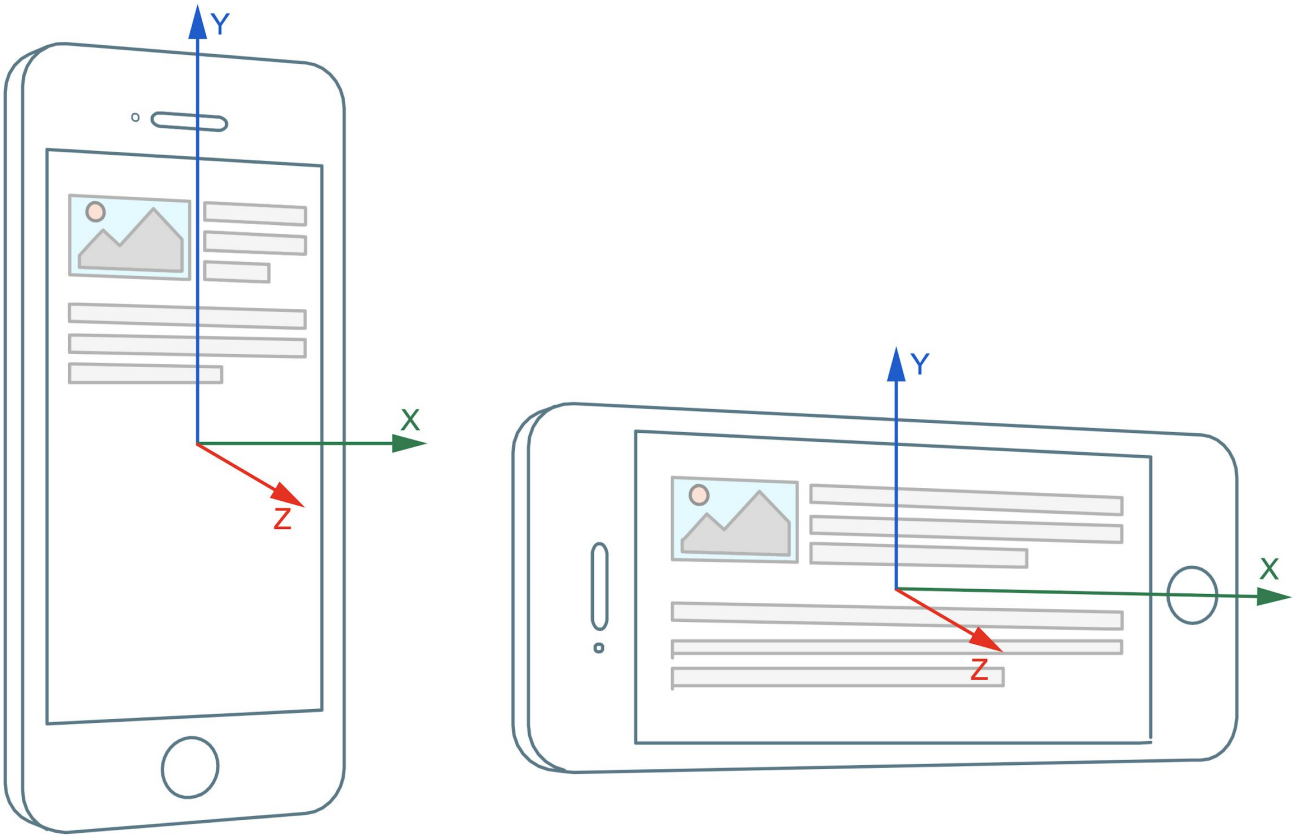


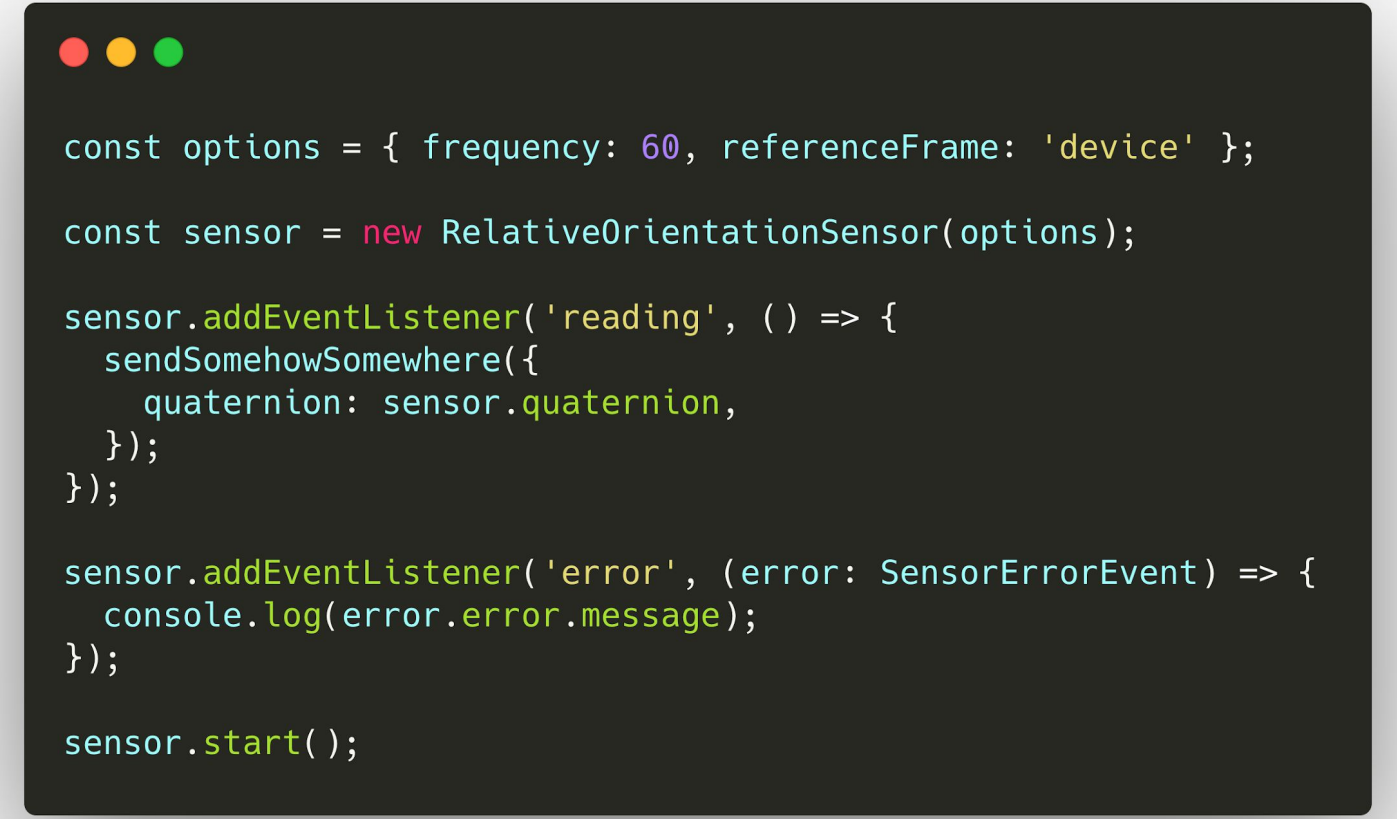
```
const options = { frequency: 60, referenceFrame: 'device' };

const sensor = new RelativeOrientationSensor(options);

sensor.addEventListener('reading', () => {
  sendSomehowSomewhere({
    quaternion: sensor.quaternion,
  });
});

sensor.addEventListener('error', (error: SensorErrorEvent) => {
  console.log(error.error.message);
});

sensor.start();
```

Device coordinate system



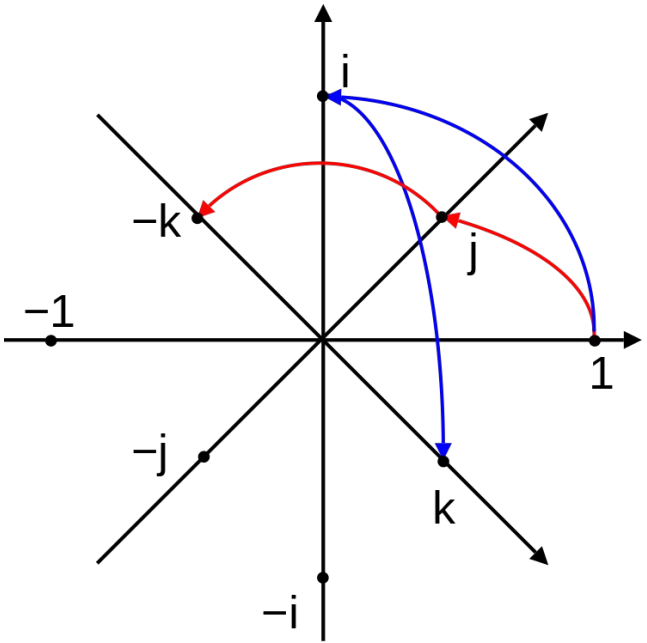
Screen coordinate system





```
const options = { frequency: 60, referenceFrame: 'device' };

const sensor = new RelativeOrientationSensor(options);

sensor.addEventListener('reading', () => {
  sendSomehowSomewhere({
    quaternion: sensor.quaternion,
  });
});

sensor.addEventListener('error', (error: SensorErrorEvent) => {
  console.log(error.error.message);
});

sensor.start();
```

Кватернион

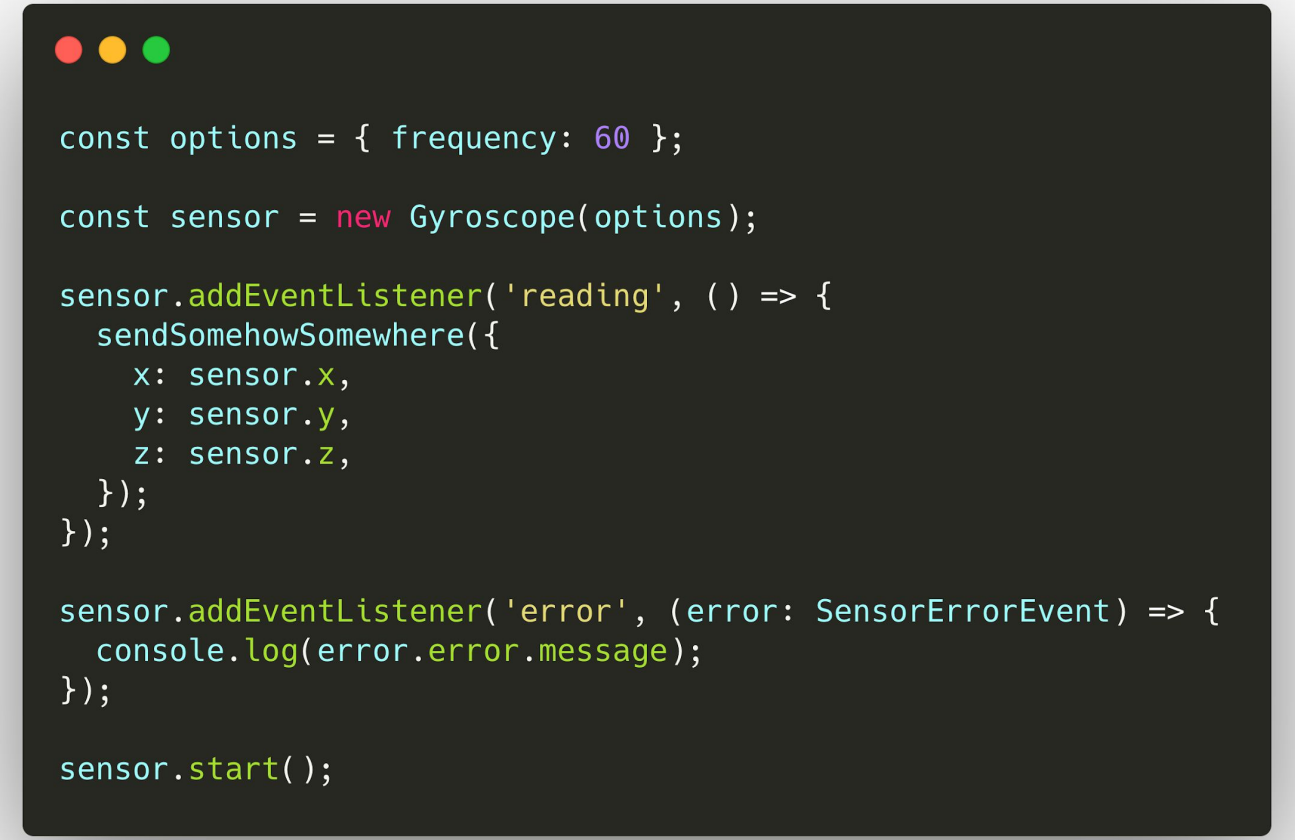


$ij = k$

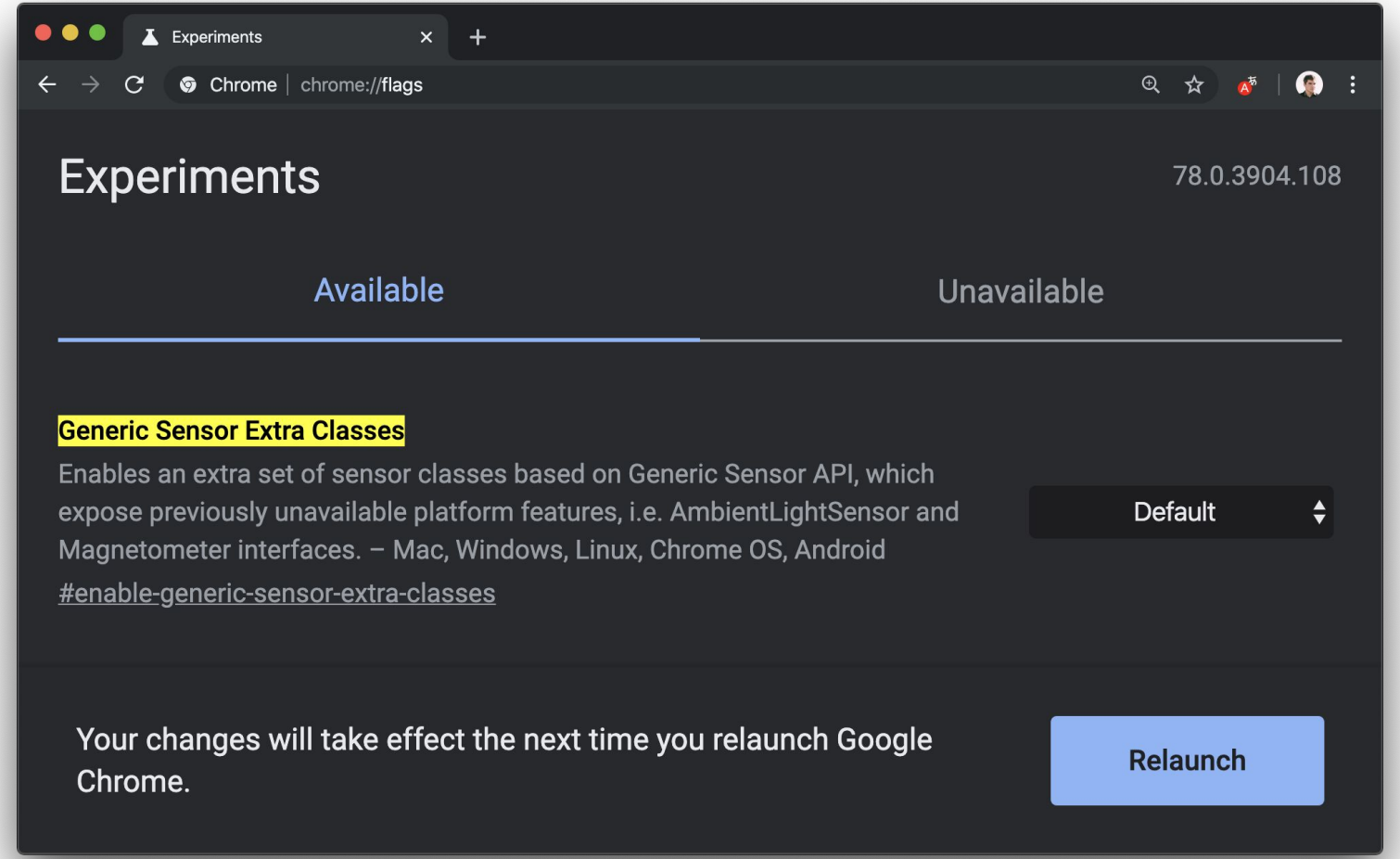
$ji = -k$

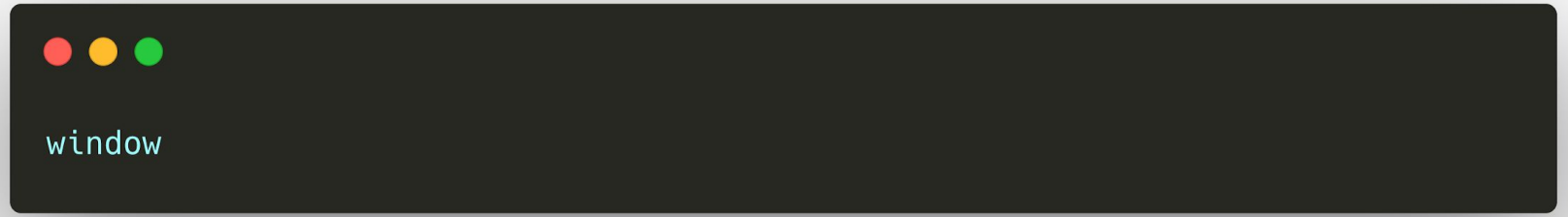
$ij = -ji$

```
const options = { frequency: 60, referenceFrame: 'device' };

const sensor = new Accelerometer(options);

sensor.addEventListener('reading', () => {
  sendSomehowSomewhere({
    x: sensor.x,
    y: sensor.y,
    z: sensor.z,
  });
});

sensor.addEventListener('error', (error: SensorErrorEvent) => {
  console.log(error.error.message);
});

sensor.start();
```



```
const options = { frequency: 60 };

const sensor = new Gyroscope(options);

sensor.addEventListener('reading', () => {
  sendSomehowSomewhere({
    x: sensor.x,
    y: sensor.y,
    z: sensor.z,
  });
});

sensor.addEventListener('error', (error: SensorErrorEvent) => {
  console.log(error.error.message);
});

sensor.start();
```


Experiments
Chrome | chrome://flags
Experiments
78.0.3904.108
Available
Unavailable
Generic Sensor Extra Classes
Enables an extra set of sensor classes based on Generic Sensor API, which expose previously unavailable platform features, i.e. AmbientLightSensor and Magnetometer interfaces. – Mac, Windows, Linux, Chrome OS, Android
#enable-generic-sensor-extra-classes
Default
Your changes will take effect the next time you relaunch Google Chrome.
Relaunch


window

```
window.addEventListener("deviceorientation", handleOrientation, true);

function handleOrientation(event) {
  const absolute = event.absolute;
  const alpha    = event.alpha;
  const beta     = event.beta;
  const gamma    = event.gamma;
}
```

```
window.addEventListener("devicemotion", handleMotionEvent, true);

function handleMotionEvent(event) {
    const x = event.accelerationIncludingGravity.x;
    const y = event.accelerationIncludingGravity.y;
    const z = event.accelerationIncludingGravity.z;
}
```

Вопросы?