

# Tackling petabytes of satellite data with 6 lines of code

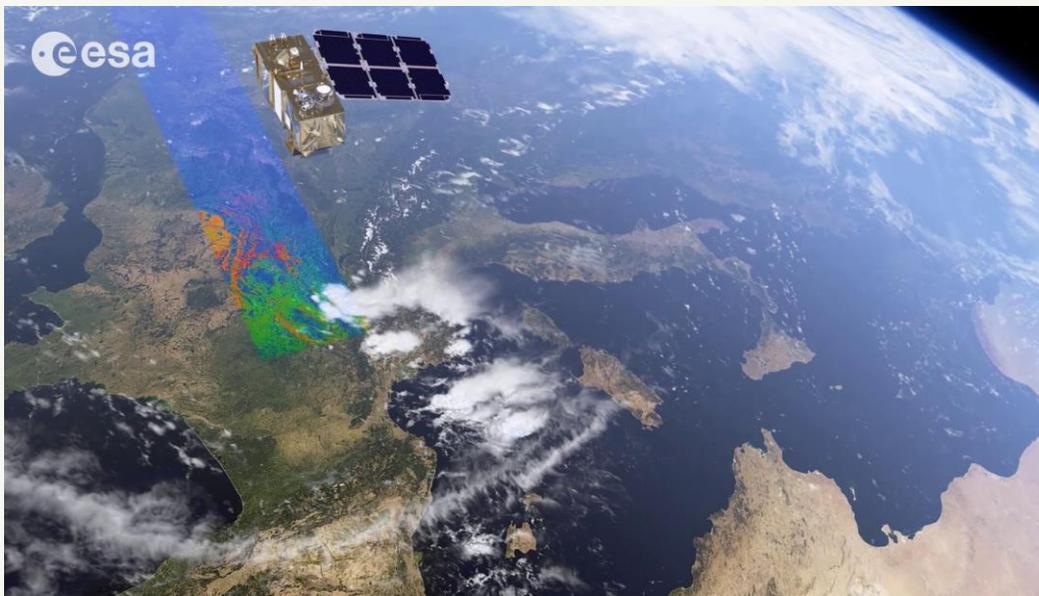
Sinergise, 2021



# Earth observation & satellite imagery

**Earth observation (EO)** is the **gathering of information** about the physical, chemical, and biological systems of the planet **via remote-sensing technologies**, ... ([Wikipedia](#))

**Satellite imagery** (also **Earth observation imagery**, **spaceborne photography**, or simply **satellite photo**) are images of Earth collected by imaging satellites ... ([Wikipedia](#))



# Sentinel Hub™ ([sentinel-hub.com](https://sentinel-hub.com))

Q: How to allow users to seamlessly and reliably access and use > 2 PB of data?

A: With Sentinel Hub™!

What is it: World-first engine for archiving, processing and distribution of Sentinel / EO data which allows users to query global archive and produce resulting data in a matter of seconds.

In short: Group of APIs from which satellite data and images can be requested

Used by:

- web applications: [Playground™](#), [EO Browser™](#), [BlueDot Water observatory](#), [Digital Twin of News](#), ...
- data science tasks: scene classification, land cover classification, cloud detection, landslide detection, parcel boundary detection, ... (a lot more on [our blog](#))

# Web applications using Sentinel Hub

<a href="#">Sentinel Playground™</a>	Simple GUI to navigate Sentinel-2 archive and play with spectral bands. Source code available on <a href="#">Github</a> .
<a href="#">EO Browser™</a>	Advanced browser for accessing complete archives of Sentinel-1,-2,-3, ESA's archive of LANDSAT-5,-7,-8, LANDSAT 8 USGS, Envisat Meris, MODIS, Proba-V and GIBS. Features advanced functionalities such as layer configuration, time-lapses and statistical analysis. Source code available on <a href="#">Github</a> .
<a href="#">BlueDot</a>	App for monitoring water bodies (lakes, rivers, ...) for floods and droughts. It uses ML and Sentinel-2 imagery.
<a href="#">Digital Twin of News</a>	Demonstration of a fusion of multi-lingual news data, Earth observation imagery, machine learning, and remote sensing techniques, which can provide important context and an interactive experience.
<a href="#">Requests builder</a>	App to help developers easily create requests for Sentinel Hub APIs.

# Data

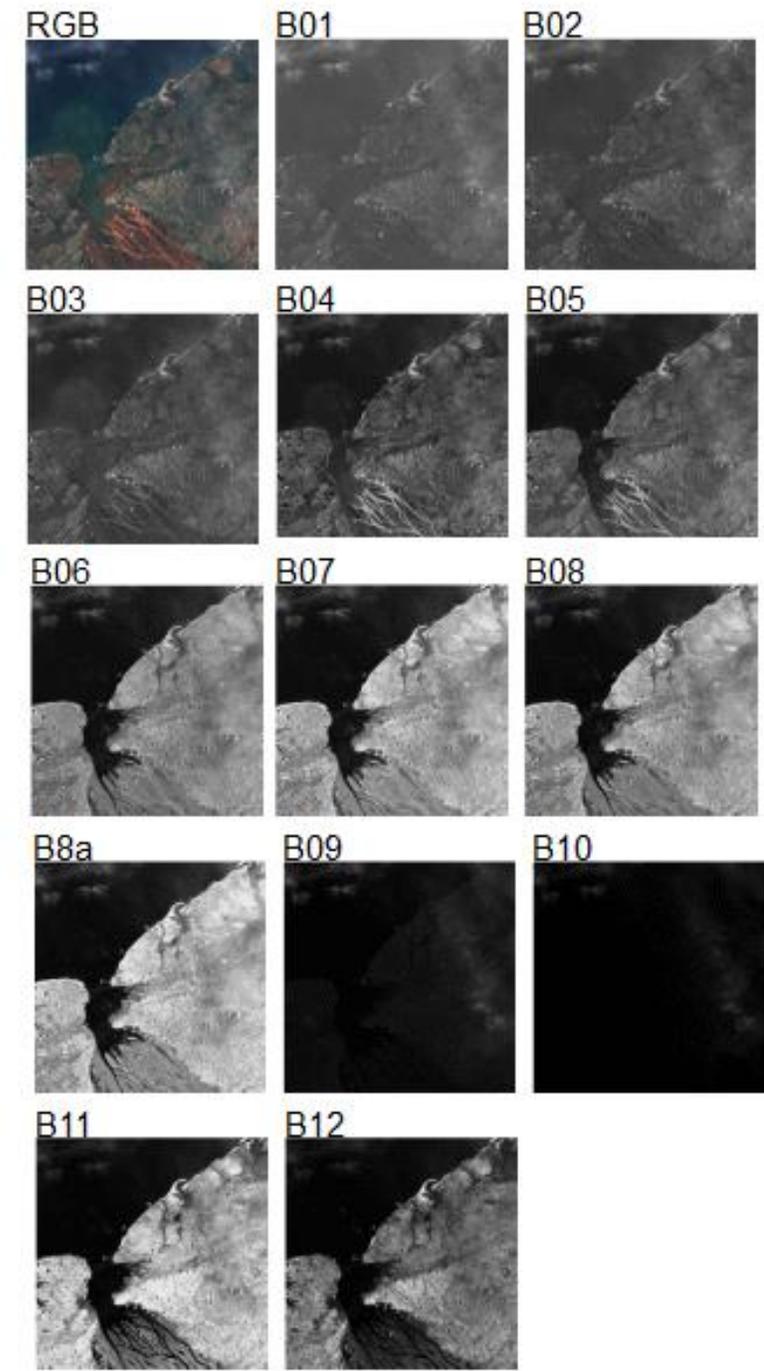
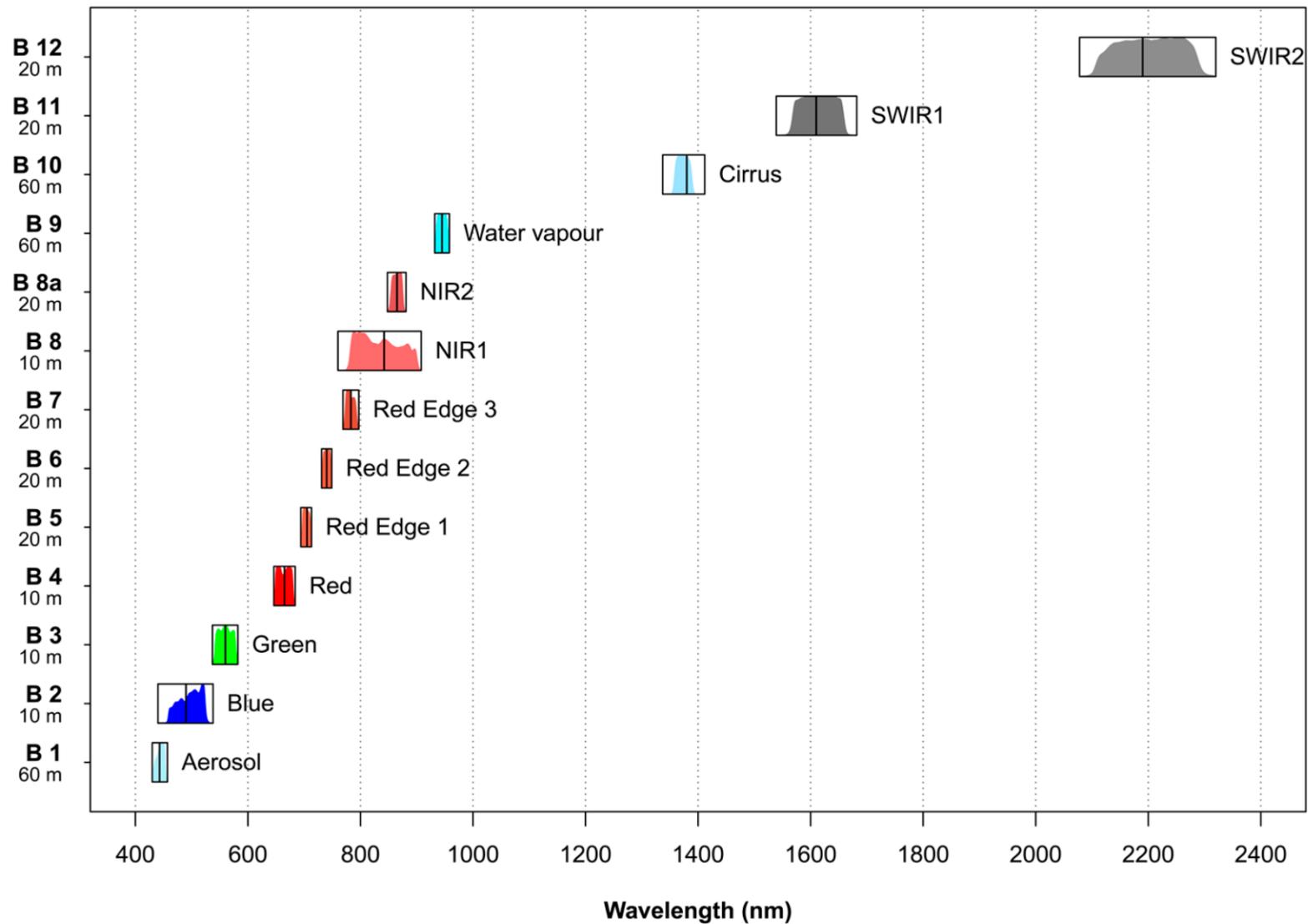
- Satellites (Sentinel-1, Sentinel-2, Sentinel-3, ...)
- Each satellite captures data in multiple bands (parts of light wavelengths)
- Each satellite captures data over some area on some date

Users define which bands from which satellites are used and how with scripts.

Most common scripts (e.g. the most natural-looking visualization – true color) are available in [SH Dashboard](#). Others are in [Custom script repository](#). You can also write your own.

End result can be an image (visualization) or "raw data" that can be used as an input for another algorithm.

# Light wavelength - bands



# How to use Sentinel Hub

- Create a [Sentinel Hub account](#)
- [OGC API](#) (old, does not need authentication, has limitations)
  - Create a configuration instance in the [Dashboard](#)
  - Add layers to the instance (use predefined or custom scripts)
- [Catalog](#), [Process](#), [Statistical](#) APIs
  - Create OAuth client in the Dashboard ([docs](#))
  - Use APIs to get the data (general info, images, statistical data)
- Ease up the work with [sentinelhub-py](#) or [sentinelhub-js](#) libraries

# Create a configuration instance

Dashboard explained [here](#)

The screenshot displays the SentinelHub Configuration Utility interface. The top navigation bar includes the SentinelHub logo, a 'Dashboard' menu, and the user name 'Ziga Cernigoj'. The main content area is titled 'Configuration Utility > mixed S1 S2'. On the left, a sidebar contains navigation options: 'Dashboard', 'Configuration Utility', 'My Collections', and 'Usage'. Below these are 'User settings', 'Billing', 'Help', and 'What's new'. At the bottom of the sidebar is a 'Collapse Sidebar' button.

The main configuration area is divided into several sections:

- Configuration instance:** A text input field containing 'mixed S1 S2'.
- Settings:** A group of toggle switches for 'Show warnings', 'Show logo', and 'Disable OGC requests'. A slider for 'Image quality' is set to 90. There are 'Add bounds' and 'Delete' buttons.
- Advanced settings:** A 'Save' button and a 'Delete' button.
- Service endpoints:** A dropdown menu for 'ID' showing '8797cb97-6ff1-4846-99b8-1cb2e1205bc8'. Below it are 'Open in Playground' and 'Copy to another account' buttons.

On the right side, there is a 'New Layer' button and a search bar. Below this is a table of layers:

Name	Id	Actions
L2A	ID: L2A	[Edit] [Delete]
S1 GRD VV	ID: S1-GRD-VV	[Edit] [Delete]
S2 fis pixels	ID: S2-FIS-PIXELS	[Edit] [Delete]
S2 true color	ID: S2-TRUE-COLOR	[Edit] [Delete]
sentinel1 test	ID: SENTINEL1-TEST	[Edit] [Delete]

The 'L2A' layer is selected, showing its configuration details:

- Source:** Sentinel-2 - L2A
- Data processing:** True color image by mapping the red, green and blue input bands. Value = B04,B03,B02 - RGB visualization. [Edit]
- Time range:** From: 2021-05-21 08:54:53, To: 2021-05-21 08:54:53
- Cloud coverage:** Slider set to 50
- Mosaic order:** Most recent

Below the configuration details are links for 'Close Preview', 'Advanced', and 'Documentation', along with a 'Delete' button. A preview window on the right shows a satellite image of a snowy landscape.

# Create a layer

Set a predefined product or custom script to the layer.

## Set custom script

Base Product	Visualization Options
TRUE_COLOR	TRUE COLOR - TRANSPARENT - True color image with transparent no-data pixels. Value = B04,B03,B02 - Output components normalized for visualization

Set Product Copy Script to Editor

```
//VERSION=3

let minVal = 0.0;
let maxVal = 0.4;

let viz = new HighlightCompressVisualizer(minVal, maxVal);

function evaluatePixel(samples) {
  let val = [samples.B04, samples.B03, samples.B02, samples.dataMask];
  return viz.processList(val);
}

function setup() {
  return {
    input: [{
      bands: [
        "dataMask",
        "B02",
```

### Custom script editor

```
1 //VERSION=3
2 // ^ this is mandatory
3 // your code
4
5 function evaluatePixel(samples) {
6   // do with the data what you want
7 }
8
9 function setup() {
10  return {
11    input: [{
12      bands: [
13        "dataMask", // dataMask = 0 when there is no data
14        "B02", // bands that you intend to use
15      ]
16    }],
17    output: {
18      bands: 4 // depends on what you return in evaluatePixel
19    }
20  };
21 }
```

Cancel Set Custom Script

# Evalsript

Evalsripts work per pixel, meaning the code in the script runs for every pixel (if a requested image is 512 \* 512 pixels big, it will execute 512 \* 512 times)

Evalsript must contain 2 functions: `setup` and `evaluatePixel`

All about evalscript [here](#)

```
//VERSION=3
function setup() { // specifies inputs and outputs
  return {
    input: ["B02", "B03", "B04"], // input bands
      // (from the collection of bands for the selected satellite)
    output: { bands: 3 } // number of output bands (3 for JPG/PNG image - R,G,B)
  };
}
function evaluatePixel(sample) { // calculates the output values for each pixel
  // sample holds values for bands that are defined in setup()
  return [2.5 * sample.B04, 2.5 * sample.B03, 2.5 * sample.B02];
}
```

# Simplifying the work

- Documentation for [OGC API](#), [Processing API](#) and [other APIs](#).
- Application [Requests builder](#) can help you build the correct requests or debug requests that are failing. Interactive examples for creating request are available [here](#).
- Python ([sentinelhub-py](#), [eo-learn](#)) libraries to ease work with SH and ML algorithms
- JavaScript ([sentinelhub-js](#)) library for frontend apps
- Custom scripts can be tested in [Requests builder](#), [Playground™](#), [EO Browser™](#)

# Javascript examples

- [Sentinel Hub Process API, Catalog API and Leaflet.js](#)
- [Sentinel Hub WMTS API and OpenLayers](#)
- Sentinel Hub WMS and Leaflet.js example can be created in Playground, if opened from the Dashboard

# Javascript example for authentication

```
async function getAuthToken() {
  const cID = encodeURIComponent(CLIENT_ID); // keep CLIENT_ID private
  const cS = encodeURIComponent(CLIENT_SECRET); // keep CLIENT_SECRET private

  const response = await fetch(
    "https://services.sentinel-hub.com/oauth/token",
    {
      method: "post",
      headers: { "Content-Type": "application/x-www-form-urlencoded" },
      body: 'grant_type=client_credentials&client_id=' + cID + '&client_secret=' + cS,
    }
  );

  const body = await response.json();
  return body["access_token"];
}

window.onload = async (event) => {
  // for example - get a token when the page is loaded
  token = await getAuthToken();
  console.log("token:", token);
};
```

# Javascript example for Catalog API

```
let payload = {
  "bbox": [36, 38, 37, 39], // EPSG:3857 or EPSG:4326
  "datetime": "2021-01-01T00:00:00Z/2021-03-29T23:59:59Z", // ISO 8601
  "collections": ["sentinel-2-l1c"], // ids are in the documentation
  "limit": 50,
  "distinct": "date", // remove to get all info, not only the dates
};
let moreResults = true;
let allDates = [];

while (moreResults) {
  const response = await fetch("https://services.sentinel-hub.com/api/v1/catalog/search", {
    method: "post",
    headers: {
      Authorization: "Bearer " + authToken,
      "Content-Type": "application/json",
      Accept: "*/*",
    },
    body: JSON.stringify(payload),
  });

  const data = await response.json();
  if (data.context.next) {
    moreResults = true;
    payload.next = data.context.next;
  } else {
    moreResults = false;
  }
  // Dates are returned in the 'features' part of the response.
  // They are ordered from the oldest to the newest date.
  allDates.push(...data.features);
}
```

# Sentinelhub-js

- `sentinelhub-js` is a JavaScript library that simplifies getting the data from Sentinel Hub in Javascript environment.
  - Available on [npm](#). Documentation is available [here](#).
  - Storybook examples showcasing some example usages are available in the [repo](#) when cloning from Github.
- Currently no simple way to use it in Javascript without using npm to install it.

# Sentinelhub-py and eolearn

- `sentinelhub-py` is a Python library that provides Python wrappers around the WMS/WCS services for image processing within the Python environment.
  - Available through PyPI or on our Github [repo](#). Documentation is available [here](#).
  - Jupyter notebooks showcasing some example usages are available in the [documentation](#) and in [example-notebooks](#) repo on Github.
- `eo-learn` is a Python library developed on top of `sentinelhub-py` for preparing the data for machine learning
- After getting the imaging data, apply machine learning Python libraries (e.g. scikit-learn, TensorFlow) to it for classification/regression/clustering tasks.

# Use cases

- Agriculture (determine when the crops are ready for harvest, which part of the field needs more fertilizer, ...)
- Land change detection (detect deforestation, expansion of urban areas, ...)
- Water resource monitoring (monitor water levels and predict water shortages, droughts, ...)
- Disaster management (assess extent of wildfires, floods, earthquakes, volcano eruptions, tsunamis, landslides, ... and plan solutions)
- Journalism, media (objective reporting on the events, disasters, ...)
- Insurance industry (assess extent of the damage, risk level, ...)
- Your idea ...

Some real-life applications are listed on [Sentinel Hub website](#)

# Ideas

## Remote sensing ideas

- Precision farming applications
- Iceberg tracking for maritime route planning
- Landslides, floods, droughts monitoring ([example](#))
- Change detection, e.g. wildfires, deforestation
- Enhancing images ([example](#))
- Visualizations of travel data
- Predict when to go to holidays where (enough snow for skiing, big waves for surfing)

# Resources 1/2

- Creating [configurations and layers](#) and [defining data processing](#)
- Creating [WMS](#), [WMTS](#) and [other](#) requests
- Creating [Processing API](#) requests
- [Using custom scripts](#)
- [Sentinelhub-py docs](#)
- [Sentinelhub-js docs](#)
- Our [Github](#)

# Resources 2/2

- [Sentinel Hub](#) (website)
- [our Github](#)
- [Documentation](#) (starting point)
- [Youtube channel](#)
- [Medium blog posts](#) (in-depth explanations, some code examples)
- [Forum](#)
- [Faq page](#)
- [Twitter account](#)
- [Educational EO stuff](#) (educational articles about Earth, EO, etc)
- [ESA Copernicus](#)
- [Remotesensing indices](#)

# Thanks

Any questions?

Don't forget to also enjoy and have a great time.  
The prize at the end is not the only important thing :)

[Open positions](#)



European Space Agency

