



⚙️ Google Earth Engine



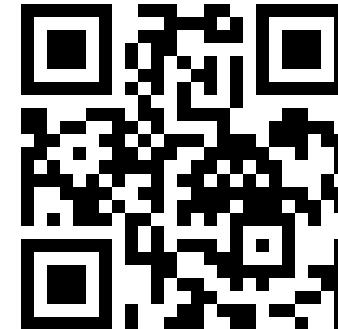
ศูนย์ภูมิภาคเทคโนโลยีอวกาศและภูมิสารสนเทศ

(ภาคเหนือ) (GISTNORTH)

ภาควิชาภูมิศาสตร์ คณะสังคมศาสตร์ มหาวิทยาลัยเชียงใหม่

เนื้อหา

- บรรยายเกี่ยวกับ GEE
- พื้นฐานการสำรวจจากระยะไกล
- ข้อมูลและเครื่องมือใช้งาน GEE
- ปฏิบัติการใช้งาน GEE
- เอกสารปฏิบัติการ
- <https://cmu.to/euOVs>





Google Earth Engine: Planetary-scale geospatial analysis for everyone

Noel Gorelick ^a , Matt Hancher ^b, Mike Dixon ^b, Simon Ilyushchenko ^b, David Thau ^b,
Rebecca Moore ^b

^a Google Switzerland, Brandschenkestrasse 110, Zurich 8002, Switzerland

^b Google Inc., 1600 Amphitheater Parkway, Mountain View, CA, 94043, USA

Received 9 July 2016, Revised 5 June 2017, Accepted 27 June 2017, Available online 6 July 2017,
Version of Record 1 December 2017.

② What do these dates mean?

Check for updates



What is Google Earth Engine?

Cloud-based Platform

Google Earth Engine (GEE)
เป็นแพลตฟอร์มบนคลาวด์
สำหรับการวิเคราะห์ภูมิ
สารสนเทศขนาดใหญ่ พัฒนา
โดย Google

Geospatial Big Data Analysis

เหมาะสมอย่างยิ่งในการประมวลผล
และวิเคราะห์ชุดข้อมูลภูมิ
สารสนเทศขนาดใหญ่ โดยมุ่งเน้นที่
ภาพถ่ายดาวเทียมและข้อมูลเชิง
พื้นที่อื่น ๆ เป็นหลัก

Streamlined Operations

GEE ดำเนินการในระบบคลาวด์ทั้งหมด จึงไม่จำเป็นต้องให้ผู้ใช้งานโหลดหรือ^{ชุดข้อมูล}
จัดเก็บชุดข้อมูลจำนวนมากไว้ในเครื่อง ทำให้กระบวนการทำงานมีประสิทธิภาพขึ้น^{ชั้น}
และลดความต้องการโครงสร้างพื้นฐาน

ข้อดีและจุดเด่นของ GEE



Cloud Computing
Leveraging

GEE ใช้เทคโนโลยี
cloud computing ของ
Google ซึ่งช่วยลด
ต้นทุนด้านโครงสร้าง
พื้นฐานได้อย่างมาก



Vast Data Repository
คลังข้อมูลขนาดใหญ่กว่า
80 petabyte จากดาวเทียม
ที่สำคัญ เช่น Landsat
Sentinel และ MODIS
รองรับการวิเคราะห์ใน
หลากหลายด้าน

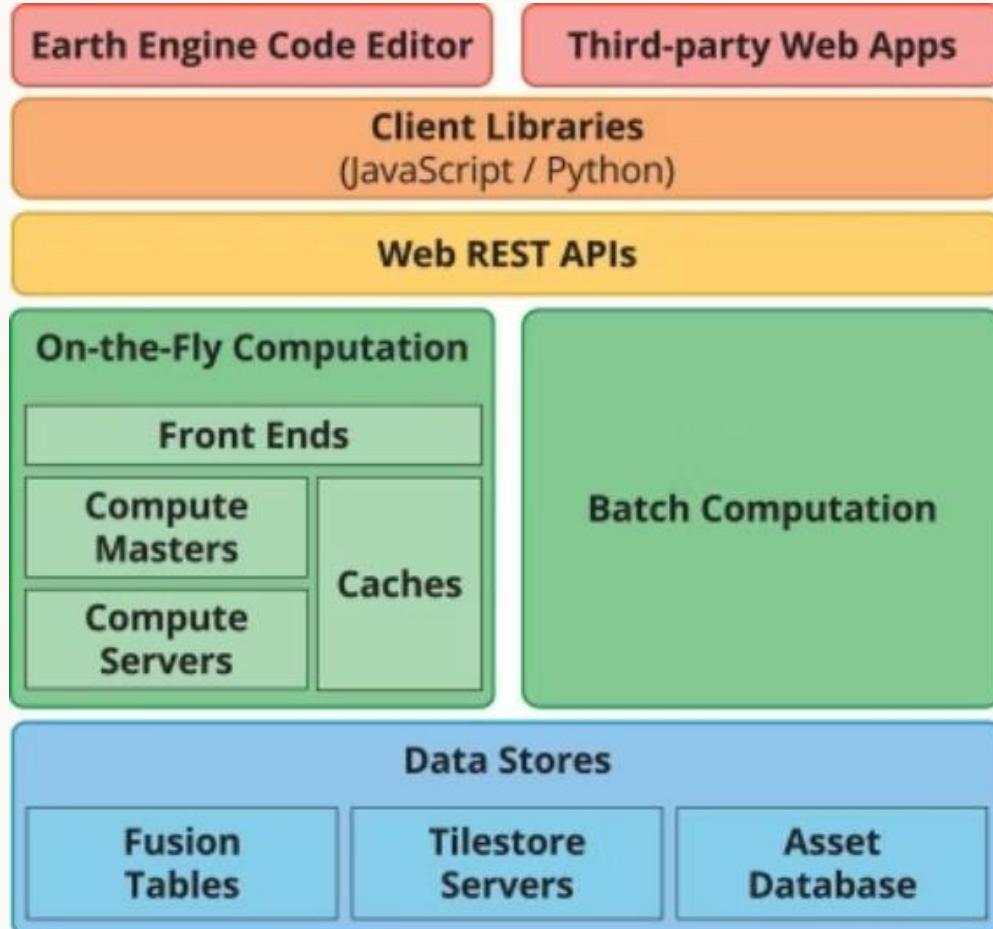


Parallel Processing
Power
แพลตฟอร์มนี้มี
ความสามารถในการ
ประมวลผลแบบขนาน ซึ่ง
ช่วยลดเวลาในการ
วิเคราะห์ชุดข้อมูลขนาด
ใหญ่ได้อย่างมาก และทำ
ให้สามารถได้ผลลัพธ์อย่าง
รวดเร็ว



Flexible API Access
รองรับการใช้งาน API ได้
ทั้งผ่าน JavaScript และ
Python ซึ่งยืดหยุ่นให้กับ
นักพัฒนา

System Architecture Overview



องค์ประกอบหลัก

โครงสร้างของ Google Earth Engine ถูกออกแบบระบบคลาวด์ของ Google รองรับการประมวลผลเชิงพื้นที่ ที่สามารถขยายได้โดยมีองค์ประกอบหลักคือ

- คลังข้อมูลขนาดใหญ่
- ระบบการประมวลผลแบบกระจาย
- API

ข้อมูลทั้งหมดถูกจัดเก็บใน

ส่วนติดต่อผู้ใช้

- JavaScript Code Editor
- Python API
- Rest API

ตัวอย่างการประยุกต์ใช้งาน GEE



ด้านการวางแผนเมืองและการติดตามการใช้ประโยชน์ที่ดิน

- วิเคราะห์การเปลี่ยนแปลงพื้นที่สีเขียว
 - การขยายตัวของเมือง
 - การตรวจสอบพื้นที่สาธารณะ
- ด้านการจัดการภัยพิบัติ
- การประเมินพื้นที่น้ำท่วม
 - การติดตามพื้นที่เสี่ยงดินถล่ม
 - ประเมินความเสียหายจากภัยแล้งหรือไฟป่า



การสำรวจระยะไกล (Remote Sensing) เป็นต้น

การเก็บข้อมูลโดยไม่สัมผัส

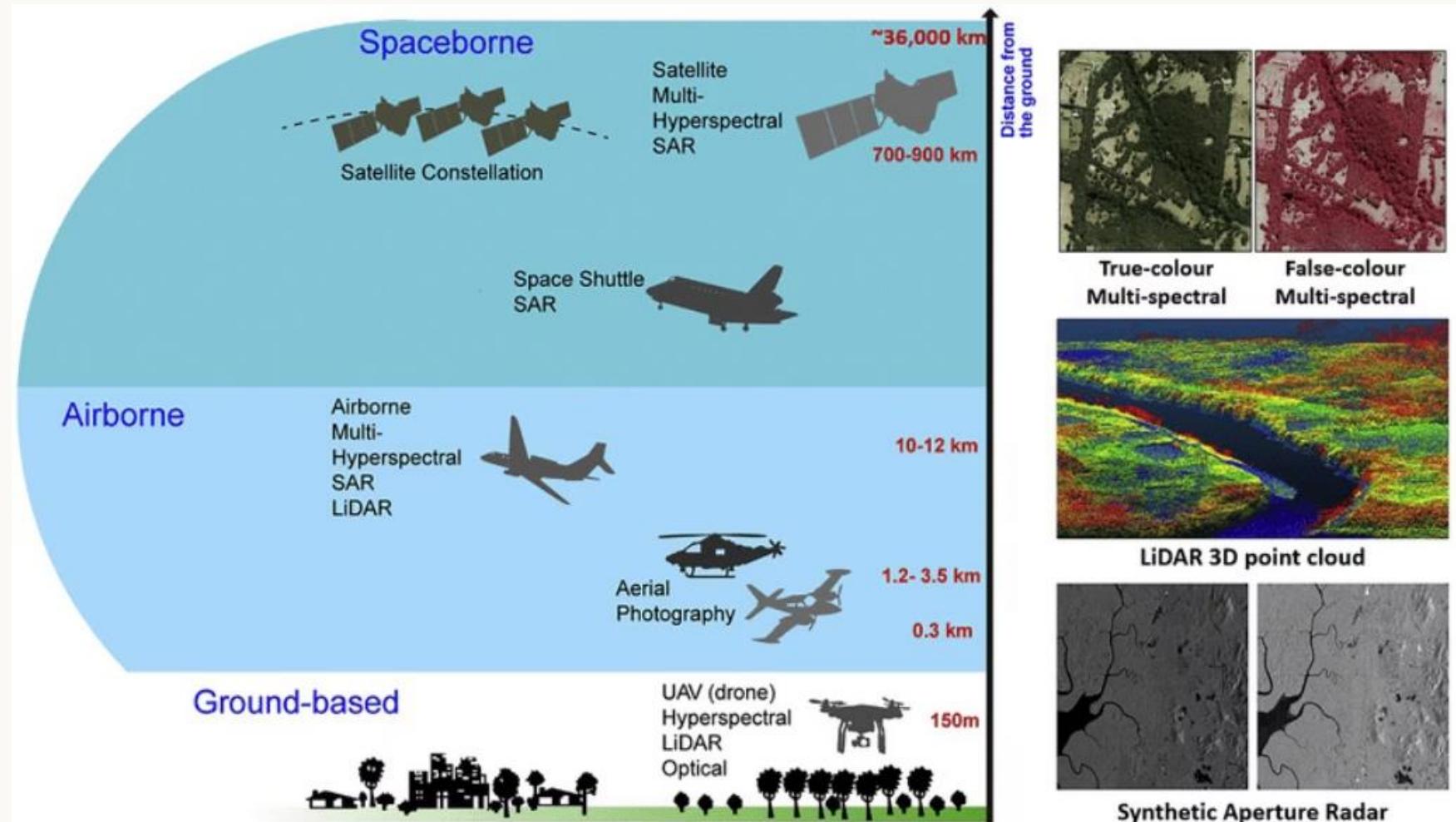
การสำรวจระยะไกล คือ การเก็บรวบรวมข้อมูลเกี่ยวกับวัตถุหรือปรากฏการณ์โดยไม่มีการสัมผัสด้วยตรง

แพลตฟอร์มการเก็บข้อมูล

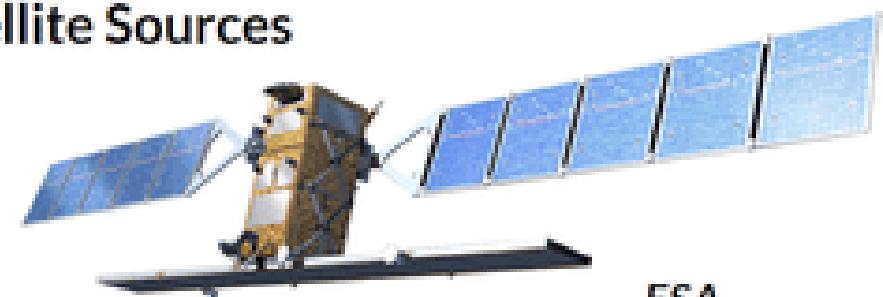
มักใช้ดาวเทียม เครื่องบิน หรือโดรนในการเก็บข้อมูล

ข้อมูลภาพจากดาวเทียม

ข้อมูลที่ได้เรียกว่า ข้อมูลภาพจากดาวเทียม หรือ ข้อมูลรีโมตเซนซิ่ง



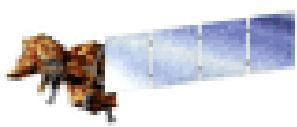
Public Satellite Sources



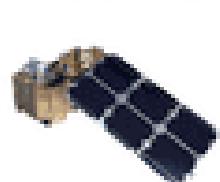
ESA
Sentinel-1A



USGS/NASA
Landsat 8



USGS/NASA
Landsat 7



ESA
Sentinel-2A/B



USGS/NASA
MODIS Terra / Aqua

Commercial Satellite Sources



Maxar
(Digital Globe)
WorldView



Airbus
Pleiades, SPOT

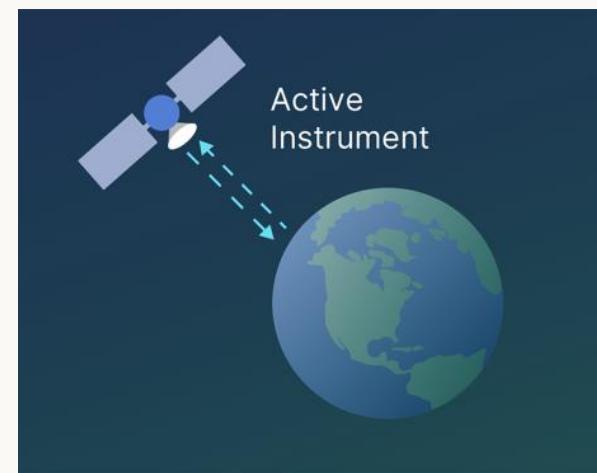
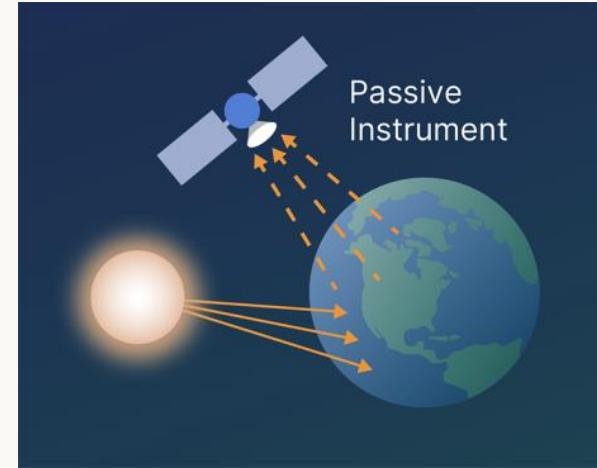


Planet
Dove, SkySat

<https://www.upstream.tech/posts/2020-07-10-remote-monitoring-glossary>

ประเภทของการสำรวจระยะไกล

- **Passive Remote Sensing** (การสำรวจแบบรับพลังงาน)
 - ดาวเทียมบันทึกพลังงานที่สะท้อนหรือแผ่ออกจากวัตถุ เช่น แสงจากดวงอาทิตย์
 - ตัวอย่างเช่น ดาวเทียม Landsat, Sentinel-2, MODIS
- **Active Remote Sensing** (การสำรวจแบบส่งพลังงาน)
 - ดาวเทียมส่งพลังงาน (เช่น ไมโครเวฟ) ลงสู่พื้นโลกแล้วบันทึกสัญญาณที่สะท้อนกลับ
 - ตัวอย่างเช่น ดาวเทียม Sentinel-1 (SAR)



องค์ประกอบสำคัญของข้อมูลภาพ จากการเทียน

1 Spatial Resolution (ความละเอียดเชิงพื้นที่)

ขนาดเล็กที่สุดของพื้นที่ที่เซ็นเซอร์สามารถแยกแยะได้ เช่น 10 เมตร หรือ 30 เมตร

2 Spectral Resolution (ความละเอียดเชิงสเปกตรัม)

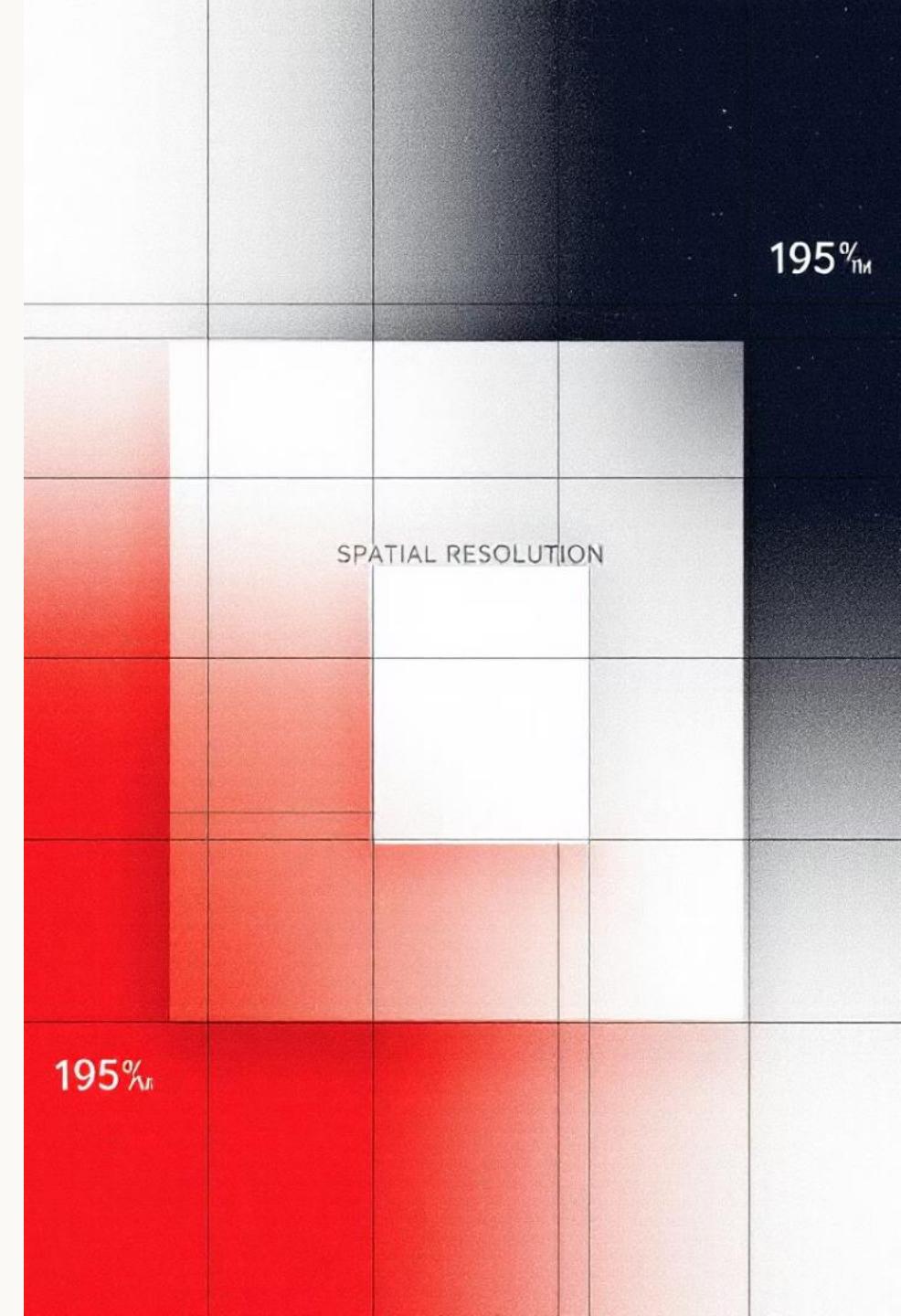
จำนวนและช่วงความยาวคลื่นที่เซ็นเซอร์บันทึกได้ เช่น RGB หรือ NIR

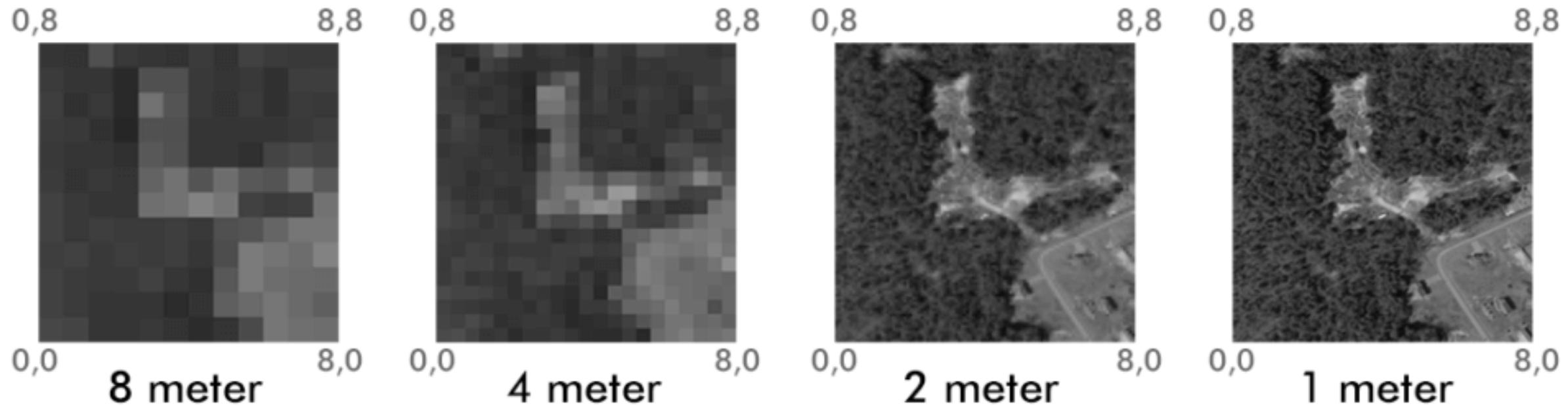
3 Temporal Resolution (ความละเอียดเชิงเวลา)

ความถี่ของการเก็บข้อมูลในตำแหน่งเดิมซ้ำๆ เช่น รายวัน หรือ รายสัปดาห์

4 Radiometric Resolution (ความละเอียดเชิงรังสี)

ความสามารถในการแยกแยะระดับความสว่าง เช่น 8-bit หรือ 16-bit





1. ความละเอียดเชิงพื้นที่

<https://mappingaround.in/digital-image-processing-in-remote-sensing/>



ภาพถ่ายดาวเทียมดิจิทัล

ใช้เซ็นเซอร์บันดาเวียมเพื่อบันทึกข้อมูลแบบดิจิทัล



Satellite image



ภาพถ่ายทางอากาศดิจิทัล

ใช้กล้องดิจิทัลบนเครื่องบินเพื่อเก็บภาพความละเอียดสูง

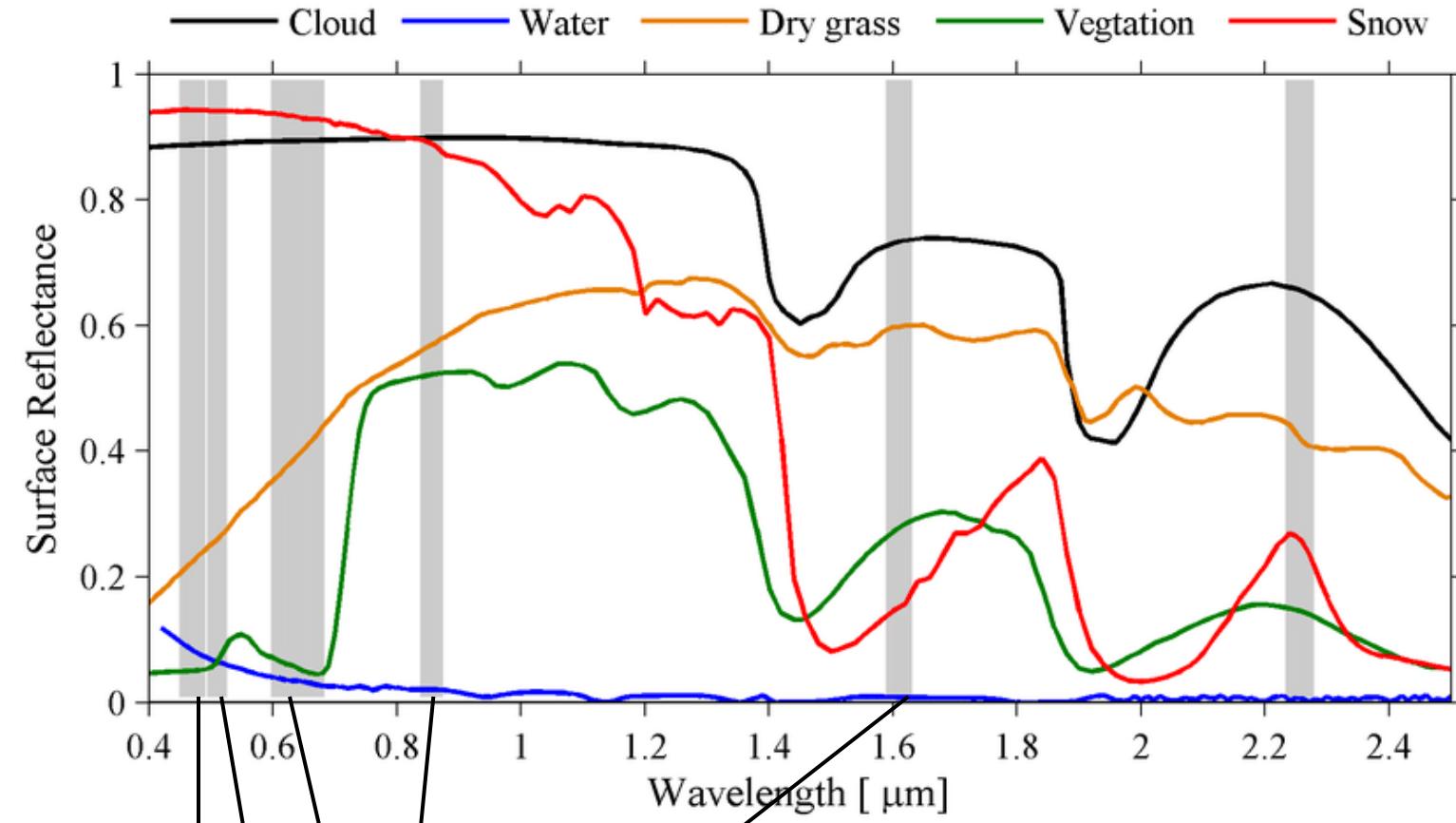
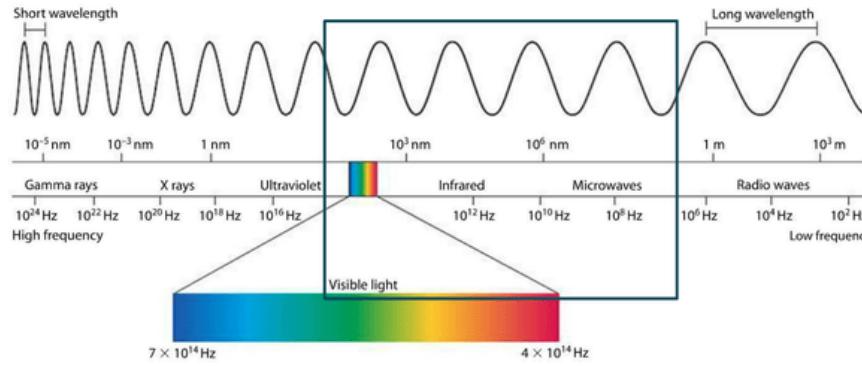


UAV image



ภาพถ่ายจากโดรน

โดรนให้ภาพระยะใกล้และมีรายละเอียดสูง
เหมาะกับพื้นที่ขนาดเล็ก

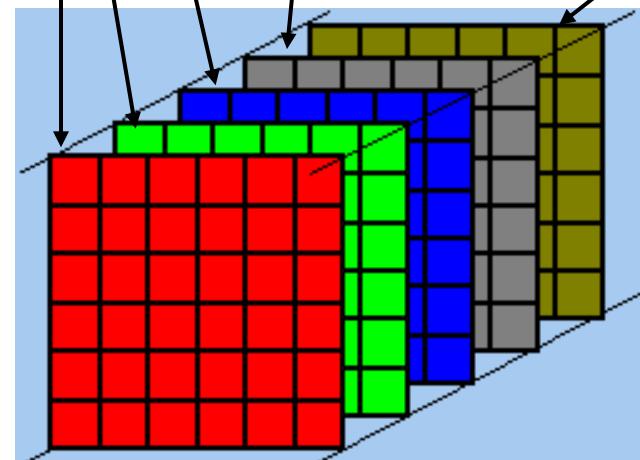


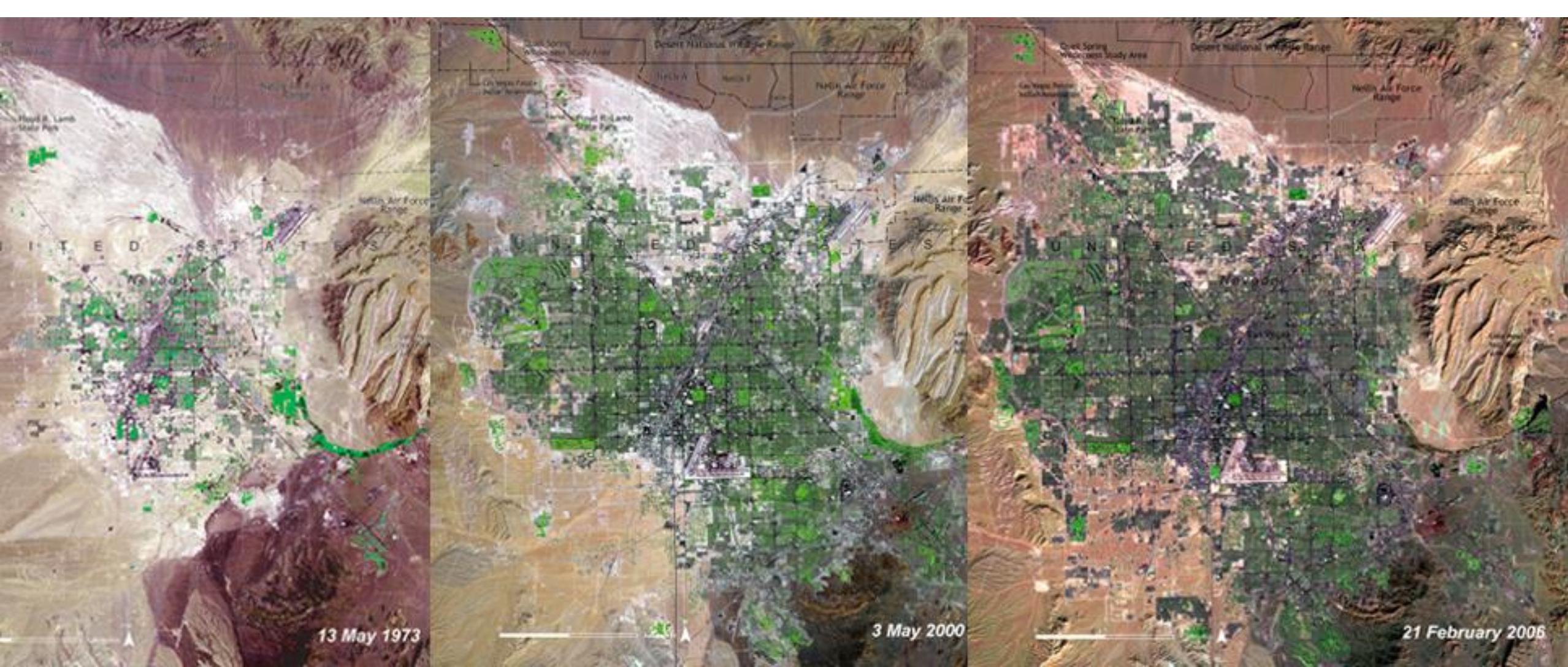
2. ความละเอียดเชิง

<https://www.upstream.tech/posts/2020-07-10-remote-monitoring-glossary>

https://www.researchgate.net/figure/Spectral-signatures-as-functions-of-wavelength-for-five-typical-surfaces-The-central_fig4_318843407

<https://mappingaround.in/digital-image-processing-in-remote-sensing/>



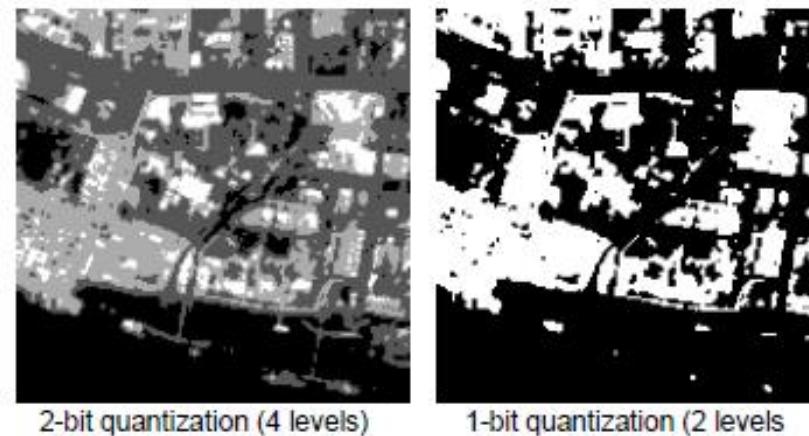
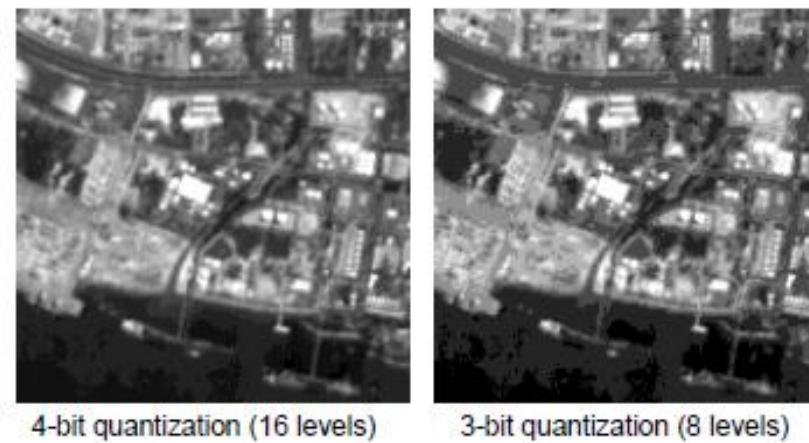


3. ความละเอียดเชิงเวลา

<https://seos-project.eu/remotesensing/remotesensing-c03-p05.html>

4. ความละเอียดเชิงรังสี

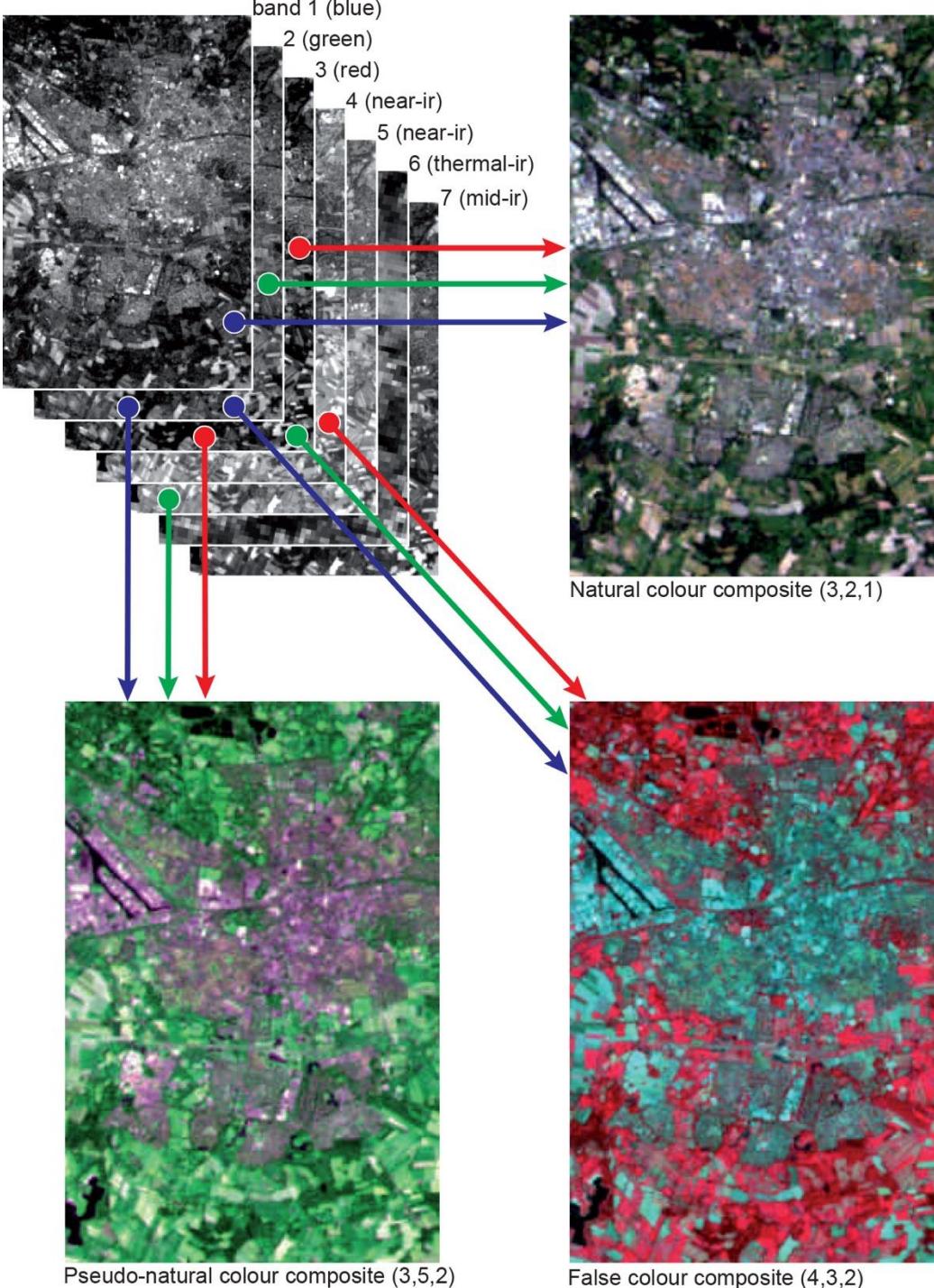
<https://mappingaround.in/digital-image-processing-in-remote-sensing/>



ตัวอย่างดาวเทียมที่มีบริการใน GEE

ดาวเทียม	ความละเอียด	ความถี่	ขนาด Bit	ตัวอย่างการใช้งาน
Landsat 8	30 เมตร	16 วัน	16-bit (DN 0–65535)	วิเคราะห์การเปลี่ยนแปลงการใช้ที่ดิน
Sentinel-2	10-20 เมตร	5 วัน	12-bit (DN 0–4095)	การประเมินสุขภาพพืชพรรณ
MODIS	250-1000 เมตร	รายวัน	12-bit / 16-bit	วิเคราะห์ไฟป่า นำท่วม
Sentinel-1 (SAR)	10 เมตร	6-12 วัน	16-bit (DN 0–65535)	วิเคราะห์พื้นที่นำท่วม พื้นที่การเกษตร

การแสดงผลข้อมูลภาพ



<https://ltb.itc.utwente.nl/509/concept/88760>

ดัชนีที่สำคัญจากภาพถ่ายดาวเทียม

- **NDVI (Normalized Difference Vegetation Index)**

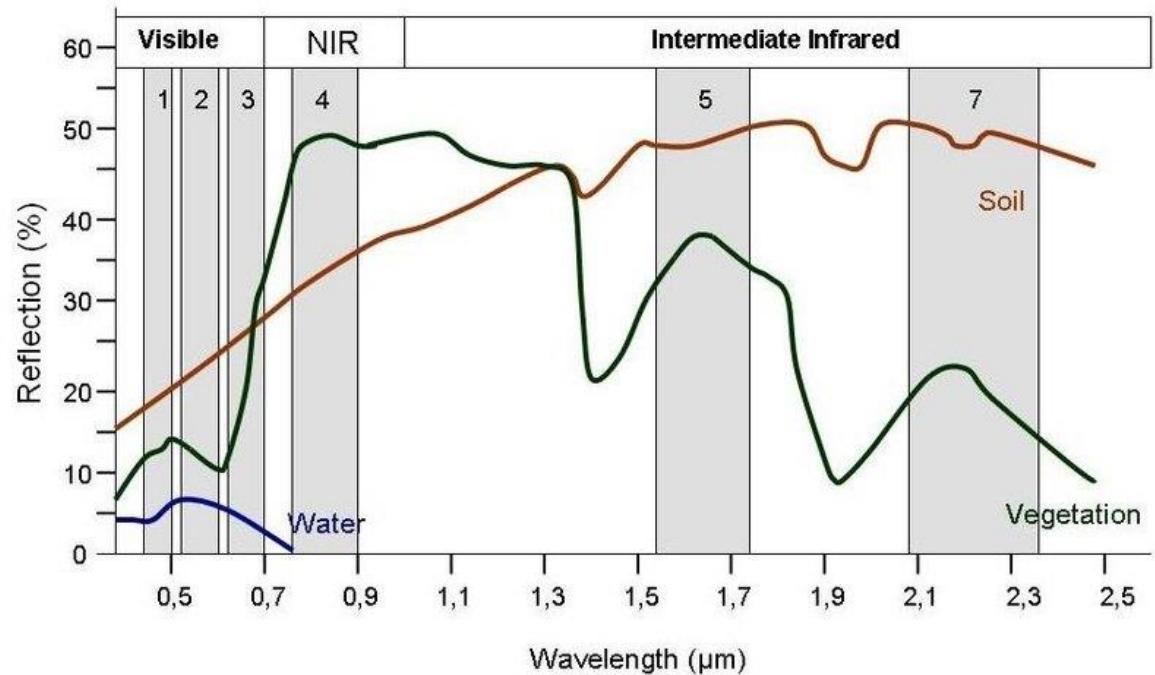
- วิเคราะห์ความหนาแน่นและสุขภาพของพืช
- คำนวณจากแบนด์ Near-Infrared (NIR) และ Red

- **NDWI (Normalized Difference Water Index)**

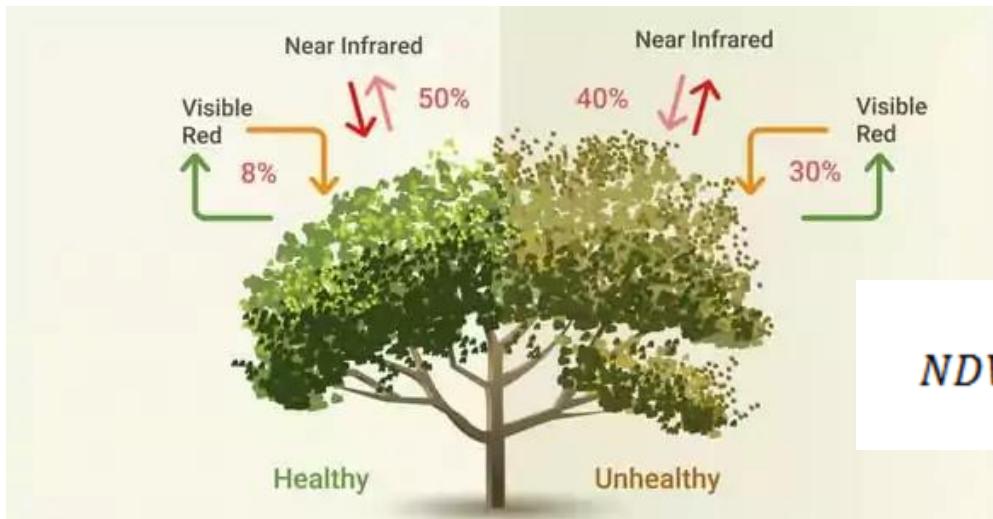
- วิเคราะห์ความชื้นในพืชและแหล่งน้ำ
- คำนวณจากแบนด์ NIR และ Shortwave-Infrared (SWIR)

- **NDBI (Normalized Difference Built-up Index)**

- วิเคราะห์พื้นที่สิ่งปลูกสร้างและเขตเมือง
- คำนวณจากแบนด์ SWIR และ NIR

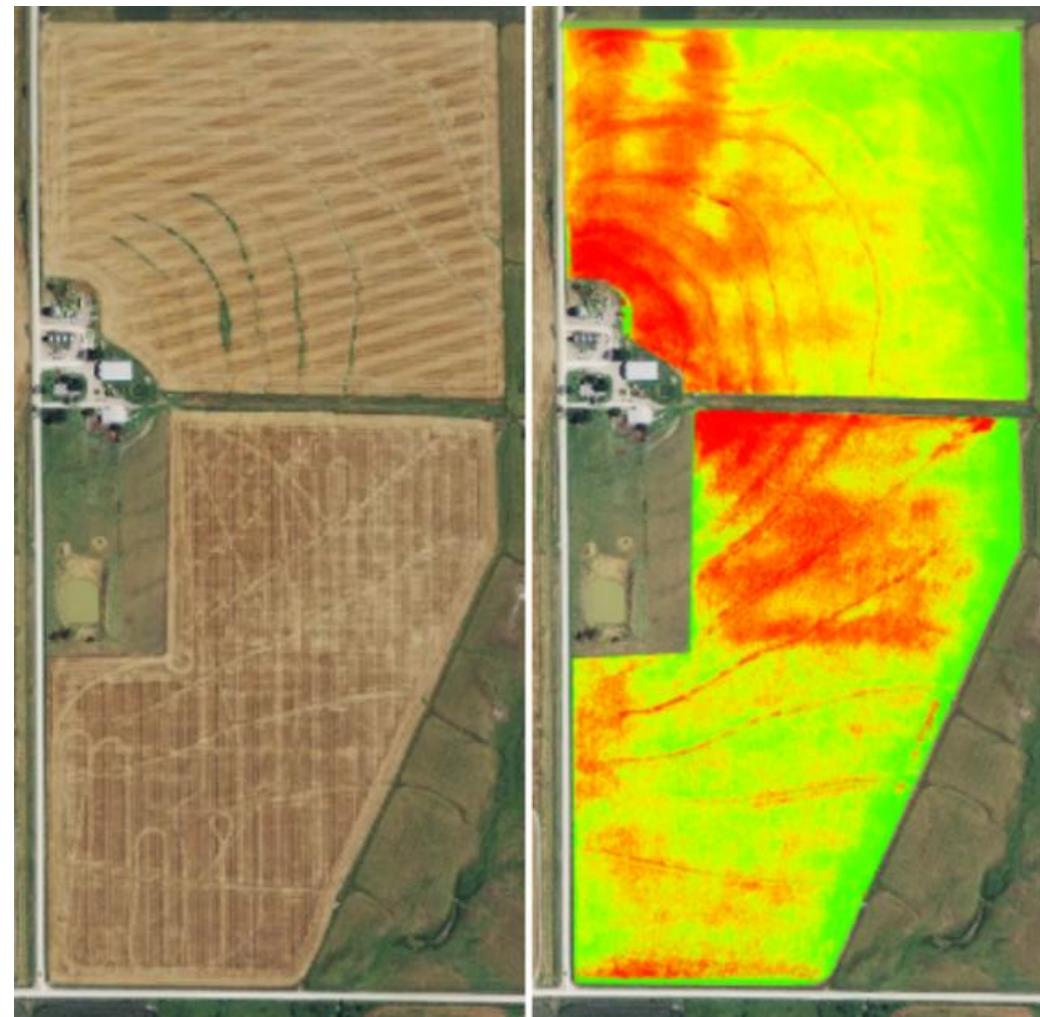


https://www.researchgate.net/figure/Spectral-signatures-of-soil-vegetation-and-water-and-spectral-bands-of-LANDSAT-7_fig2_335175625



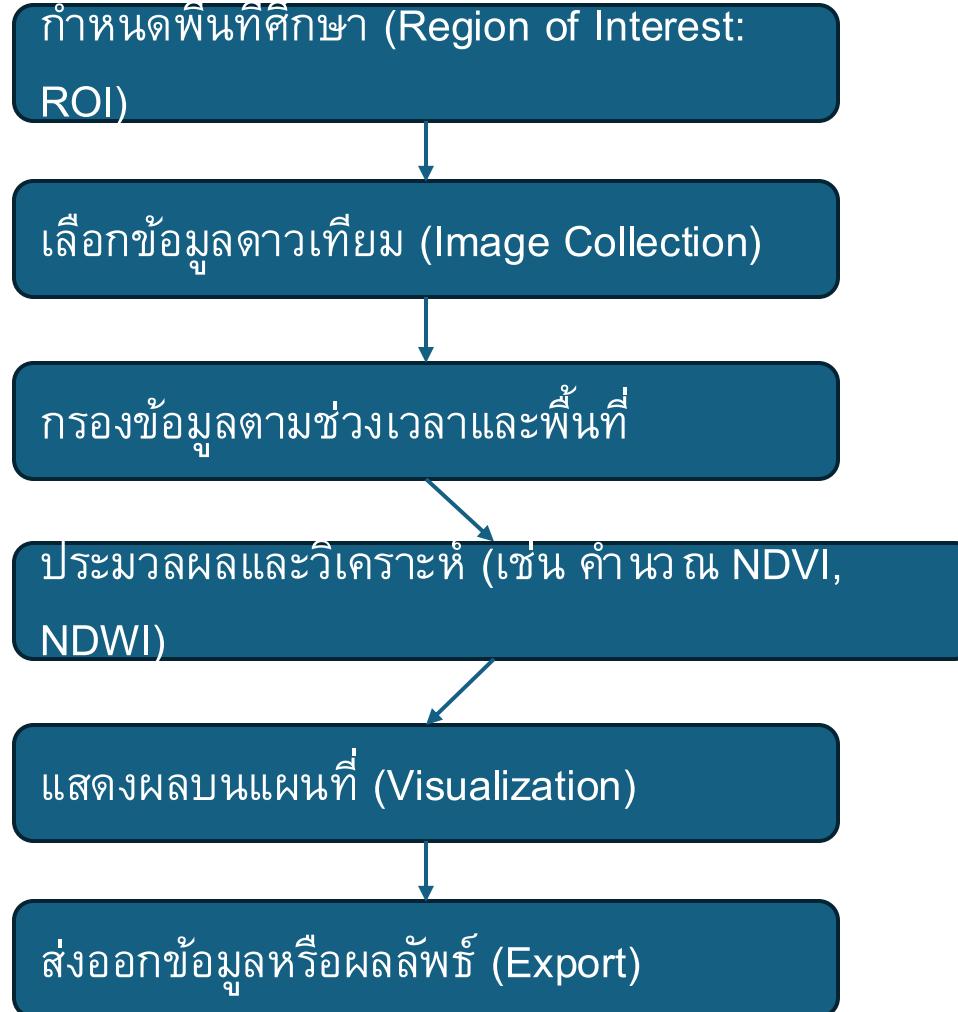
$$NDVI = \frac{(NIR - Red)}{(NIR + Red)}$$

<https://eos.com/blog/ndvi-faq-all-you-need-to-know-about-ndvi/>



<https://botlink.com/blog/ndvi-vs-false-ndvi-whats-better-for-analyzing-crop-health>

ขั้นตอนทั่วไปในการวิเคราะห์ข้อมูล Remote Sensing ด้วย GEE



วิธีในการใช้งาน Google Earth Engine

Google Earth Engine (GEE) ให้ผู้ใช้งานเข้าถึงและวิเคราะห์ข้อมูลภูมิสารสนเทศได้ผ่าน 3 ช่องทางหลัก ได้แก่

1. ใช้งานผ่าน JavaScript API (GEE Code Editor)

- **รูปแบบ:** Web-based IDE (Integrated Development Environment)
- **วิธีการใช้งาน:** เข้าผ่านเว็บไซต์ code.earthengine.google.com
- **จุดเด่น:**
 - ใช้งานง่าย เหมาะสำหรับผู้เริ่มต้น
 - ไม่ต้องติดตั้งโปรแกรมเพิ่มเติม
 - รองรับการเขียนโค้ด JavaScript โดยตรงในเบราว์เซอร์

ตัวอย่างการใช้งานผ่าน GEE Code Editor

Scripts Docs Assets New Script * Get Link Save Run Reset Apps Inspector Console Tasks

ee.Array
ee.Blob
ee.Classifier
ee.Clusterer
ee.ConfusionMatrix
ee.Date
ee.DateRange
ee.Dictionary
ee.ErrorMargin

```
75 // 9. Display the results
76 Map.centerObject(roi);
77 Map.addLayer(comp1, {bands: ['B4','B3','B2'], min:0, max:3000}, 'True Color 1', false)
78 Map.addLayer(classified1, {min:0, max:3, palette: palette}, 'Classified Jan-Mar', true)
79
80 Map.addLayer(comp2, {bands: ['B4','B3','B2'], min:0, max:3000}, 'True Color 2', false)
81 Map.addLayer(classified2, {min:0, max:3, palette: palette}, 'Classified Jul-Sep', true)
82
83
84 // 10. Optional: accuracy assessment for period 1
85 var trainTest1 = samples1.randomColumn('rnd', 42);
86 var split = 0.7;
```

Use print(...) to write to this console.

FeatureCollection pr... JSON

Confusion matrix (1st... JSON

[[21,0,0,0],[0,1,1,0,... JSON

Overall accuracy: JSON

0.9989462592202318

Geometry Imports Layers Map Satellite

Google

GISTNORTH

Keyboard shortcuts Map data ©2025 Google 2 km Terms Report a map error

2. ใช้งานผ่าน Python API (Client Library)

- **รูปแบบ:** Python Library (earthengine-api)
- **วิธีการใช้งาน:** ติดตั้งผ่านคำสั่ง pip (pip install earthengine-api)
- **จุดเด่น:**
 - เหมาะกับงานวิจัยหรือโครงการที่ต้องการประมวลผลข้อมูลด้วยภาษา Python
 - สามารถใช้งานร่วมกับ Python libraries อื่นๆ เช่น NumPy, Pandas, GeoPandas
 - รองรับการใช้งานใน Jupyter Notebook, Colab และสคริปต์ Python อื่นๆ

ตัวอย่างการใช้งานผ่าน Python API

```
import ee  
ee.Initialize()  
  
image = ee.Image('COPERNICUS/S2/20240101T000000')  
url = image.getThumbURL({'min':0, 'max':3000, 'bands':['B4','B3','B2']})  
print(url)
```

3. ใช้งานผ่าน REST API (เชื่อมต่อโดยตรง)

- **รูปแบบ:** เชื่อมต่อโดยตรงกับเซิร์ฟเวอร์ GEE ผ่าน REST API
- **วิธีการใช้งาน:** ส่ง HTTP requests ตรงไปยัง GEE servers
- **จุดเด่น:**
 - เหมาะกับนักพัฒนาที่ต้องการสร้างแอปพลิเคชันหรือระบบที่ทำงานอัตโนมัติร่วมกับ GEE
 - มีความยืดหยุ่นสูงในการใช้งานร่วมกับระบบต่าง ๆ

ตัวอย่างการใช้งานผ่าน REST API

```
POST https://earthengine.googleapis.com/v1/projects/your-project-id/assets:exportImage
{
  "expression": {
    "image": {
      "assetId": "COPERNICUS/S2/20240101T000000"
    }
  },
  "fileFormat": "GEO_TIFF",
  "region": {...}
}
```

เปรียบเทียบการใช้งานทั้ง 3 วิธี

วิธีใช้งาน	ความง่าย	ความยืดหยุ่น	เหมาะสมกับ
GEE Code Editor	ง่ายที่สุด	ปานกลาง	เรียนรู้/ใช้งานทั่วไป
Python API	ปานกลาง	สูง	งานวิจัยและงานวิเคราะห์ข้อมูลสูง
REST API	ยากที่สุด	สูงสุด	การพัฒนาแอปพลิเคชันและระบบอัตโนมัติ

วิธีเข้าใช้งาน

- เข้าไปที่เว็บไซต์ earthengine.google.com
- คลิก "Sign Up" หรือ "Get Started"
- ใช้บัญชี Google (เช่น Gmail) ในการลงทะเบียน
- ระบุวัตถุประสงค์ชัดเจน เช่น ใช้ในการศึกษา วิจัย หรือประโยชน์สาธารณะ

The screenshot shows the Google Earth Engine interface. At the top, there's a navigation bar with 'Google Earth Engine' and a search bar. Below it, a menu bar has 'Scripts' selected, followed by 'Docs', 'Assets', and various action buttons: 'Get Link', 'Save', 'Run', 'Reset', and 'Apps'. The main area is divided into several sections:

- Script Editor:** Displays a portion of a JavaScript script:

```
1 // A UI to interactively filter a collection  
2 // from the results, display it with a vari  
3  
4 // The namespace for our application. All  
5 var app = {};  
6  
7 /** Creates the UI panels. */  
8 app.createPanels = function() {  
9   /* The map to use for this app. */  
10  app.map = new ui.Map();  
11  
12  /* The introduction section. */  
13  app.intro = {  
14    panel: ui.Panel([
```
- Map View:** Shows a satellite map of a coastal area with roads labeled '40D' and '69E'. A large blue polygon covers a portion of the landmass. A legend at the bottom right includes 'Layers', 'Map', and 'Satellite' buttons.
- Controls:** On the left, there are zoom controls (+, -, ⌂). Below the map, there are dropdown menus for 'Select an image' (set to 'LC08_026042_20170515') and 'Select a visualization' (set to 'Atmospheric (B7/B6/B5)').
- Description:** A text box below the visualization dropdown says: "Coast lines and shores are well-defined. Vegetation appears blue."

การเปิดใช้งาน Google Cloud Project

หลังสมัคร GEE ต้องเปิดใช้งาน Google Cloud

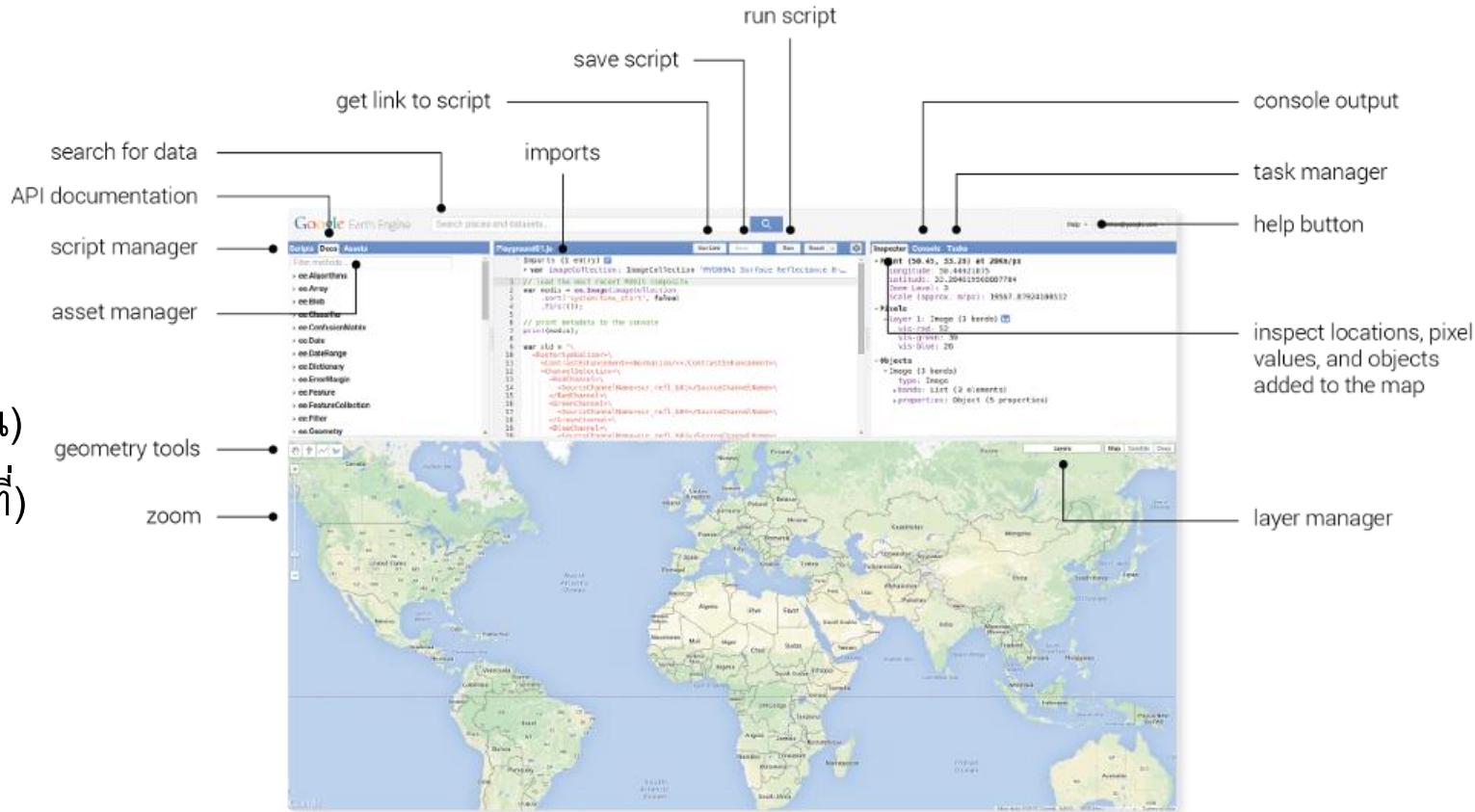
- สร้าง Project ใน Google Cloud และเลือก non-commercial use
- อธิบายเหตุผลในการใช้งานให้ชัดเจนเพื่อขออนุมัติการใช้งานจาก Google

หลังจากได้รับอนุมัติแล้ว สามารถเข้าที่เว็บไซต์:

- URL: code.earthengine.google.com
- ล็อกอินด้วยบัญชี Google ที่อนุมัติ
- แนะนำการเปิดใช้งานครั้งแรกและการเลือก Cloud Project ที่สร้างไว้

ส่วนประกอบของ GEE Code Editor

- Panel การใช้งาน
 - **Scripts** (สคริปต์)
 - **Docs** (เอกสารอ้างอิง)
 - **Assets** (จัดการข้อมูลส่วนตัว)
 - **Console** (แสดงผลลัพธ์และแจ้งเตือน)
 - **Inspector** (ตรวจสอบข้อมูลบนแผนที่)
 - **Tasks** (จัดการงานที่ส่งประมวลผล)



ข้อมูลใน GEE

Dataset ใน GEE คืออะไร?

- **Dataset (ชุดข้อมูล)** ใน Google Earth Engine (GEE) คือ กลุ่มข้อมูลเชิงพื้นที่ (**Geospatial Data**) ที่ถูกจัดเก็บและพร้อมใช้งานบนระบบคลาวด์ กลุ่มข้อมูลเชิงพื้นที่ (**Geospatial Data**) คือ:
 - ข้อมูลที่เกี่ยวข้องกับตำแหน่งหรือพื้นที่บนพื้นผิวโลก
 - โดยทั่วไปแสดงในรูปแบบแรสเตอร์ (Raster) เช่น ภาพถ่ายดาวเทียม, DEM หรือรูปแบบเวกเตอร์ (Vector) เช่น ขอบเขตพื้นที่, ถนน, แม่น้ำ
 - ข้อมูลเหล่านี้มีการอ้างอิงตำแหน่งที่ชัดเจนบนพื้นผิวโลกด้วยพิกัดทางภูมิศาสตร์ (Latitude และ Longitude)

ตัวอย่างชุดข้อมูลที่มีใน GEE:

- ภาพถ่ายดาวเทียม (เช่น Landsat, Sentinel, MODIS)
- ข้อมูลสภาพภูมิอากาศและอากาศ (Climate and Weather)
- ข้อมูลภูมิประเทศและระดับความสูง (Terrain & DEM)
- ข้อมูลการปักคุลพื้นที่และการใช้ที่ดิน (Land Cover & Land Use)
- ชุดข้อมูลเหล่านี้พร้อมใช้งานทันที โดยผู้ใช้ไม่จำเป็นต้องดาวน์โหลดหรือเก็บข้อมูลไว้ในเครื่อง
- สามารถนำไปใช้เคราะห์ผ่าน GEE Code Editor หรือ API ต่างๆ ได้ทันที เพื่อสนับสนุนงานวิจัย การศึกษา และการบริหารจัดการเชิงพื้นที่

Imagery Dataset (ข้อมูลภาพจากดาวเทียม)

- คือชุดข้อมูลภาพถ่ายที่ได้จากการถ่ายดาวเทียมและเซนเซอร์ต่างๆ
- ตัวอย่างดาวเทียมสำคัญ:
 - Landsat:** ความละเอียด 30 เมตร ตั้งแต่ปี 1972 ถึงปัจจุบัน
 - Sentinel-2:** ความละเอียด 10-20 เมตร เหมาะสมกับการวิเคราะห์ NDVI
 - Sentinel-1:** ระบบเรดาร์ (SAR) ความละเอียด 10 เมตร ทะลุเมฆได้ดี
- ที่มา: developers.google.com



Landsat Collection

ชื่อดาวเทียม	เซ็นเซอร์หลัก	ข้อมูลที่ได้	ความละเอียดเชิงพื้นที่	ความถี่การผ่าน	ข้อมูลย้อนหลัง	ตัวอย่างการใช้งาน
Landsat 4	TM (Thematic Mapper)	Multispectral 7 แบนด์ (VIS, NIR, SWIR, Thermal)	30 ม. (TIR: 120 ม.)	16 วัน	ปี 1982–1993	การใช้ที่ดิน, NDVI, การเปลี่ยนแปลงพื้นที่ป่า
Landsat 5	TM	เหมือน Landsat 4	30 ม. (TIR: 120 ม.)	16 วัน	ปี 1984–2013	ป่าไม้, การเกษตร, วิเคราะห์แนวโน้มหลายศักราช
Landsat 7	ETM+ (Enhanced Thematic Mapper Plus)	Multispectral + Panchromatic + Thermal	30 ม. (Pan: 15 ม.)	16 วัน	ปี 1999–ปัจจุบัน	การจำแนกพื้นที่, ตรวจสอบเมือง, นำทั่วม
Landsat 8	OLI (Optical) + TIRS (Thermal Infrared Sensor)	11 แบนด์: VIS, NIR, SWIR, Thermal, Cirrus	30 ม. (Pan: 15 ม., TIR: 100 ม.)	16 วัน	ปี 2013–ปัจจุบัน	NDVI, LST, การวิเคราะห์พื้นที่เพาะปลูก, คุณภาพน้ำ
Landsat 9	OLI-2 + TIRS-2	เหมือน Landsat 8 แต่ปรับปรุงคุณภาพเชิงรังสี	30 ม. (Pan: 15 ม., TIR: 100 ม.)	16 วัน	ปี 2021–ปัจจุบัน	งานวิจัยสิ่งแวดล้อม, วางแผนพื้นที่เกษตร, เปรียบเทียบหล่ายช่วงเวลา

Collection 2

Landsat Collection 2, the second major reprocessing effort on the Landsat archive, resulted in several data product improvements that applied advancements in data processing and algorithm development.



Landsat 9 OLI-2/TIRS-2

2021–Present



Landsat 8 OLI/TIRS

2013–Present



Landsat 7 ETM+

1999–2021



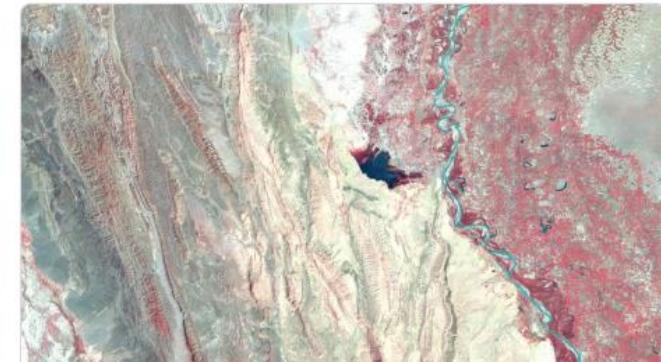
Landsat 5 TM

1984–2012



Landsat 4 TM

1982–1993



Landsat 1-5 MSS

1972–1999

USGS Landsat 9 Level 2, Collection 2, Tier 1



Dataset Availability

2021-10-31T00:00:00Z–2025-05-29T23:52:08.473000Z

Dataset Provider

USGS

Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC09/C02/T1_L2")
```

Revisit Interval

16 Days

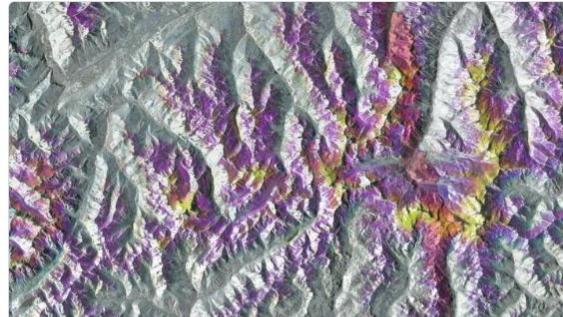
Tags

cfmask, cloud, fmask, global, l9sr, landsat, lasrc, lc09, lst, reflectance
satellite-imagery, sr, usgs

Description	Bands	Image Properties	Terms of Use				
Pixel Size							
30 meters							
Bands							
Name	Units	Min	Max	Scale	Offset	Wavelength	Description
SR_B1		1	65455	2.75e-05	-0.2	0.435-0.451 µm	Band 1 (ultra blue, coastal aerosol) surface reflectance
SR_B2		1	65455	2.75e-05	-0.2	0.452-0.512 µm	Band 2 (blue) surface reflectance
SR_B3		1	65455	2.75e-05	-0.2	0.533-0.590 µm	Band 3 (green) surface reflectance
SR_B4		1	65455	2.75e-05	-0.2	0.636-0.673 µm	Band 4 (red) surface reflectance
SR_B5		1	65455	2.75e-05	-0.2	0.851-0.879 µm	Band 5 (near infrared) surface reflectance
SR_B6		1	65455	2.75e-05	-0.2	1.566-1.651 µm	Band 6 (shortwave infrared 1) surface reflectance
SR_B7		1	65455	2.75e-05	-0.2	2.107-2.294 µm	Band 7 (shortwave infrared 2) surface reflectance
SR_QA_AEROSOL	Aerosol attributes						

Sentinel Collections

ข้อมูลจาก ดาวเทียม Sentinel คือชุดข้อมูลจาก โครงการ Copernicus ของสหภาพยุโรป (EU) และ องค์การอวกาศยุโรป (ESA) ซึ่งออกแบบมาเพื่อการตรวจสอบโลก (Earth Observation) เพื่อสนับสนุนการจัดการทรัพยากรธรรมชาติ, ภัยพิบัติ, การเกษตร, ป่าไม้, การใช้ที่ดิน, คุณภาพอากาศ ฯลฯ



Sentinel-1 SAR GRD: C-band Synthetic Aperture Radar

Data availability: 2014 – Present

The Sentinel-1 mission provides data from a dual-polarization C-band Synthetic Aperture Radar (SAR) instrument. SAR instruments are capable of acquiring meaningful data in all weather conditions (even clouds) during daytime and nighttime. Sentinel-1 data is used across many domains, including maritime activity, sea-ice mapping, humanitarian aid, crisis response, and forest management.



Sentinel-2 MSI: Multispectral Instrument

Data availability: 2015 – Present

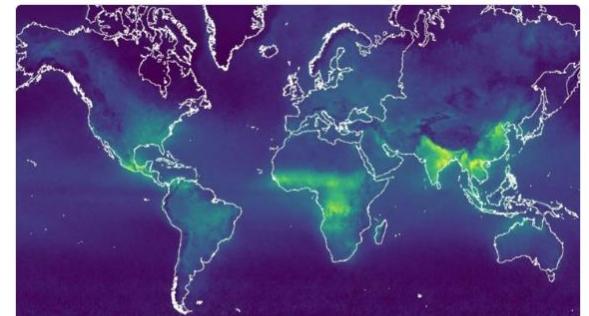
The Sentinel-2 mission collects high-resolution multispectral imagery useful for a broad range of applications, including monitoring of vegetation, soil and water cover, land cover change, as well as humanitarian and disaster risk.



Sentinel-3 OLCI EFR: Ocean and Land Color Instrument

Data availability: 2016 – Present

The Sentinel-3 instrument provides systematic measurements of the planet's oceans, land, ice, and atmosphere, including the temperature, color and height of the sea surface as well as the thickness of sea ice.



Sentinel-5 TROPOMI: TROPOspheric Monitoring Instrument

Data availability: 2018 – Present

The Sentinel-5 Precursor mission collects data useful for assessing air quality, including concentrations of: ozone, methane, formaldehyde, aerosol, carbon monoxide, nitrogen oxide, and sulphur dioxide.

ประเภทของดาวเทียม Sentinel

Sentinel satellites มีหลายดวงและแต่ละดวงมีภารกิจเฉพาะที่แตกต่างกัน

ชื่อดาวเทียม	เซ็นเซอร์หลัก	ข้อมูลที่ได้	ความละเอียด	ความถี่การผ่าน	การใช้งาน
Sentinel-1	Synthetic Aperture Radar (SAR-C)	เรดาร์ (VV, VH)	~10 เมตร	6–12 วัน	ตรวจน้ำท่วม, การเคลื่อนตัวของดิน
Sentinel-2	Multispectral Instrument (MSI)	ภาพถ่ายทาง ช่วงคลื่น (13 bands)	10–60 เมตร	5 วัน (S2A+S2B)	การเกษตร, ป่าไม้, NDVI
Sentinel-3	OLCI, SLSTR, SRAL	อุณหภูมิผิวน้ำ, ความสูงน้ำทะเล	300–1,200 เมตร	~1–2 วัน	มหาสมุทร, ภูมิอากาศ
Sentinel-5P	TROPOMI	มลพิษอากาศ (NO ₂ , O ₃ , SO ₂ , CO)	5.5 x 3.5 กม. Q3TNORTH	1 วัน	ติดตามคุณภาพ อากาศ

Sentinel-2

- **Sentinel-2** ชิ้งเป็นดาวเทียมบันทึกภาพหลายช่วงคลื่น (Multispectral)



Surface Reflectance

Level-2A orthorectified atmospherically corrected surface reflectance.

Dataset availability: 2017-03-28 – Present



Top-of-Atmosphere Reflectance

Level-1C orthorectified top-of-atmosphere reflectance.

Dataset availability: 2015-06-27 – Present

Harmonized Sentinel-2 MSI: MultiSpectral Instrument, Level-2A (SR)



Dataset Availability

2017-03-28T00:00:00Z–2025-05-31T19:55:03.306000Z

Revisit Interval

5 Days

Dataset Provider

European Union/ESA/Copernicus

Tags

copernicus esa eu msi reflectance satellite-imagery sentinel sr

Earth Engine Snippet

```
ee.ImageCollection("COPERNICUS/S2_SR_HARMONIZED")
```

Description	Bands	Image Properties	Terms of Use				
Bands							
Name Units Min Max Scale Pixel Size Wavelength Description							
B1		0.0001	60 meters		443.9nm (S2A) / 442.3nm (S2B)	Aerosols	
B2		0.0001	10 meters		496.6nm (S2A) / 492.1nm (S2B)	Blue	
B3		0.0001	10 meters		560nm (S2A) / 559nm (S2B)	Green	
B4		0.0001	10 meters		664.5nm (S2A) / 665nm (S2B)	Red	
B5		0.0001	20 meters		703.9nm (S2A) / 703.8nm (S2B)	Red Edge 1	
B6		0.0001	20 meters		740.2nm (S2A) / 739.1nm (S2B)	Red Edge 2	
B7		0.0001	20 meters		782.5nm (S2A) / 779.7nm (S2B)	Red Edge 3	
B8		0.0001	10 meters		835.1nm (S2A) / 833nm (S2B)	NIR	

Terrain / Elevation Dataset (ข้อมูลระดับความสูง)

- ข้อมูลระดับความสูงและภูมิประเทศ (Digital Elevation Model: DEM)
- ตัวอย่างข้อมูล DEM ยอดนิยม:
 - **SRTM**: ความละเอียด 30 เมตร ระหว่าง -60° ถึง $+60^{\circ}$ ละติจูด
 - **ASTER GDEM**: ความละเอียด 30 เมตร ครอบคลุมทั่วโลก
- ประโยชน์ใช้: วิเคราะห์การไฟลของน้ำ, การประเมินความเสี่ยงดินถล่ม
- ที่มา: developers.google.com



พื้นฐานภาษา JavaScript สำหรับ GEE

- GEE Code Editor รองรับภาษา JavaScript เป็นหลัก
- ใช้เขียนสคริปต์วิเคราะห์ข้อมูลภูมิสารสนเทศผ่าน Cloud
- เขียนคำสั่งเชิงวิเคราะห์ (NDVI, การกรองภาพ, การทำ mask) ได้ง่าย
- มีฟังก์ชันเฉพาะของ Earth Engine (เช่น ee.Image, ee.Geometry)

แนวคิด Client-side vs Server-side

- **Client-side:** ทำงานในเบราว์เซอร์ เช่น print(), Map.addLayer()
- **Server-side:** ประมวลผลข้อมูลขนาดใหญ่บนเซิร์ฟเวอร์ Google
- ตัวแบบ ee.Object จะทำงานผ่าน Server-side
- ต้องเข้าใจความแตกต่างเพื่อไม่ให้เกิด error จากการจัดการข้อมูลผิดฝั่ง

ตัวอย่าง Server-side

```
var image = ee.Image('COPERNICUS/S2/20240101T000000');
```

ตัวอย่าง Client-side

```
Map.addLayer(image);  
print(image);
```

- การใช้ `print()` กับ `ee.Object` จะต้องให้เซิร์ฟเวอร์ดึงค่ากลับมาที่เบราว์เซอร์
- การพยายามใช้ค่าจาก `Server-side` มาประมวลผลใน `Client-side` โดยตรงจะทำให้เกิดข้อผิดพลาด เช่น `image.getInfo()` ต้องใช้กับข้อมูลขนาดเล็กเท่านั้น
- หลักการ: ควรเขียนสคริปต์ให้ทำงานอยู่ ผู้ดูแล Server-side ให้มากที่สุดเพื่อความรวดเร็ว

การประการตัวแปร

- ใช้ var ในการประกาศตัวแปรใน JavaScript
- var city = 'Chiang Mai'; var value = 100;
- ตั้งชื่อตัวแปรตามหลัก Camel Case เช่น
ndviValue
- หลีกเลี่ยงการใช้คำส่วน (reserved words)

Google JavaScript Style Guide

Revision 2.93

Aaron Whyte
Bob Jervis
Dan Pupius
Erik Arvidsson
Fritz Schneider
Robby Walker

Each style point has a summary for which additional information is available by toggling the accompanying arrow button that looks this way: ▶. You may toggle all summaries with the big arrow button:

 Toggle all summaries

Table of Contents

JavaScript Language Rules	var Constants Semicolons Nested functions Function Declarations Within Blocks Exceptions Custom exceptions Standards features Wrapper objects for primitive types Multi-level prototype hierarchies Method and property definitions delete Closures eval() with() this for-in loop Associative Arrays Multiline string literals Array and Object literals Modifying prototypes of builtin objects Internet Explorer's Conditional Comments
JavaScript Style Rules	Naming Custom toString() methods Deferred initialization Explicit scope Code formatting Parentheses Strings Visibility (private and protected fields) JavaScript Types Comments Providing Dependencies With goog.provide Compiling Tips and Tricks

https://google.github.io/styleguide/javascriptguide.xml#JavaScript_Language_Rules

ประเภทข้อมูลพื้นฐาน

- Number: ตัวเลข เช่น 10, 3.14
- String: ข้อความ เช่น 'GEE', "เชียงใหม่"
- Boolean: true, false
- Array: กลุ่มข้อมูล เช่น [1, 2, 3]
- Object: เช่น {name: 'Thailand', area: 513000}

การใช้ Array และ Object

- **Object** ใช้เก็บข้อมูลในรูปแบบ key-value เหมาะสำหรับเก็บข้อมูลที่เกี่ยวข้องกัน

```
var province = {  
    name: "Chiang Mai",  
    population: 1700000,  
    area: 20107  
};  
print(province.name); // Chiang Mai
```

- **Array** ใช้เก็บชุดของข้อมูลที่มีลำดับ เช่น รายชื่อจังหวัด หรือค่าฝนในแต่ละวัน

```
var provinces = ["Chiang Mai", "Lamphun", "Chiang Rai"];  
print(provinces[0]); // Chiang Mai
```

ฟังก์ชัน (Function)

- ฟังก์ชันคือกลุ่มคำสั่งที่นำกลับมาใช้ซ้ำได้
- สามารถรับพารามิเตอร์และคืนค่าผลลัพธ์

```
function add(x, y) {  
    return x + y;  
}  
var sum = add(10, 5);  
print("Sum is", sum);
```

- ใน GEE นิยมใช้ฟังก์ชันกับการ map ข้อมูล หรือคำนวณค่าต่าง ๆ

```
var collection = ee.ImageCollection('COPERNICUS/S2')
  .filterDate('2024-01-01', '2024-01-31')
  .filterBounds(ee.Geometry.Point(100.5, 13.7));

var ndviCollection = collection.map(function(image) {
  var ndvi = image.normalizedDifference(['B8', 'B4'])
    .rename('NDVI');
  return image.addBands(ndvi);
});
```

การใช้ If-Else

- ใช้เพื่อตรวจสอบเงื่อนไขและตัดสินใจว่าจะทำคำสั่งใดต่อไป

```
var score = 85;
if (score >= 80) {
    print('A');
} else {
    print('B หรือ ต่ำกว่า');
}
```

- ใน GEE server-side ใช้เมธอดเปรียบเทียบเช่น .gt(), .lt() และ if-else ธรรมดा

```
var mask = image.normalizedDifference(["B8", "B4"]).gt(0.5)
```

Object-Oriented Programming (OOP)

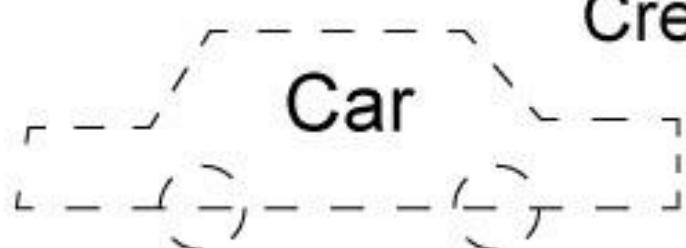
แนวคิดของ OOP

- เป็นการเขียนโปรแกรมโดยใช้แนวคิด “วัตถุ” (Object) ซึ่งรวมข้อมูล (properties) และฟังก์ชัน (methods) ไว้ภายในวัตถุเดียวกัน
- ส่งเสริมแนวทางการเขียนโค้ดที่สามารถนำกลับมาใช้ซ้ำ (reuse) ได้ง่าย และโครงสร้างโค้ดมีความชัดเจน องค์ประกอบของ Object

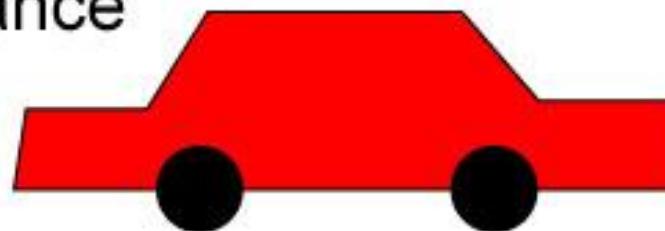
1. Properties (คุณสมบัติ) เป็นข้อมูลที่เก็บอยู่ใน object

2. Methods (เมธอด) เป็นฟังก์ชันที่ทำงานกับข้อมูลภายใน object นั้น

Class



Create an instance



Properties	Methods - behaviors
color	start()
price	backward()
km	forward()
model	stop()

Property values	Methods
color: red	start()
price: 23,000	backward()
km: 1,200	forward()
model: Audi	stop()

- ตัวอย่างการสร้าง Object และ Method

```
var landsat = {  
    satellite: "Landsat 8",  
    bands: ["B2", "B3", "B4", "B5"],  
    getNIR: function() {  
        return this.bands[3]; // คืนค่า NIR band  
    },  
    info: function() {  
        return this.satellite + " with bands: " + this.bands.join(", ");  
    }  
};  
  
print("NIR band:", landsat.getNIR());  
print("Satellite info:", landsat.info());
```

Method Chaining:

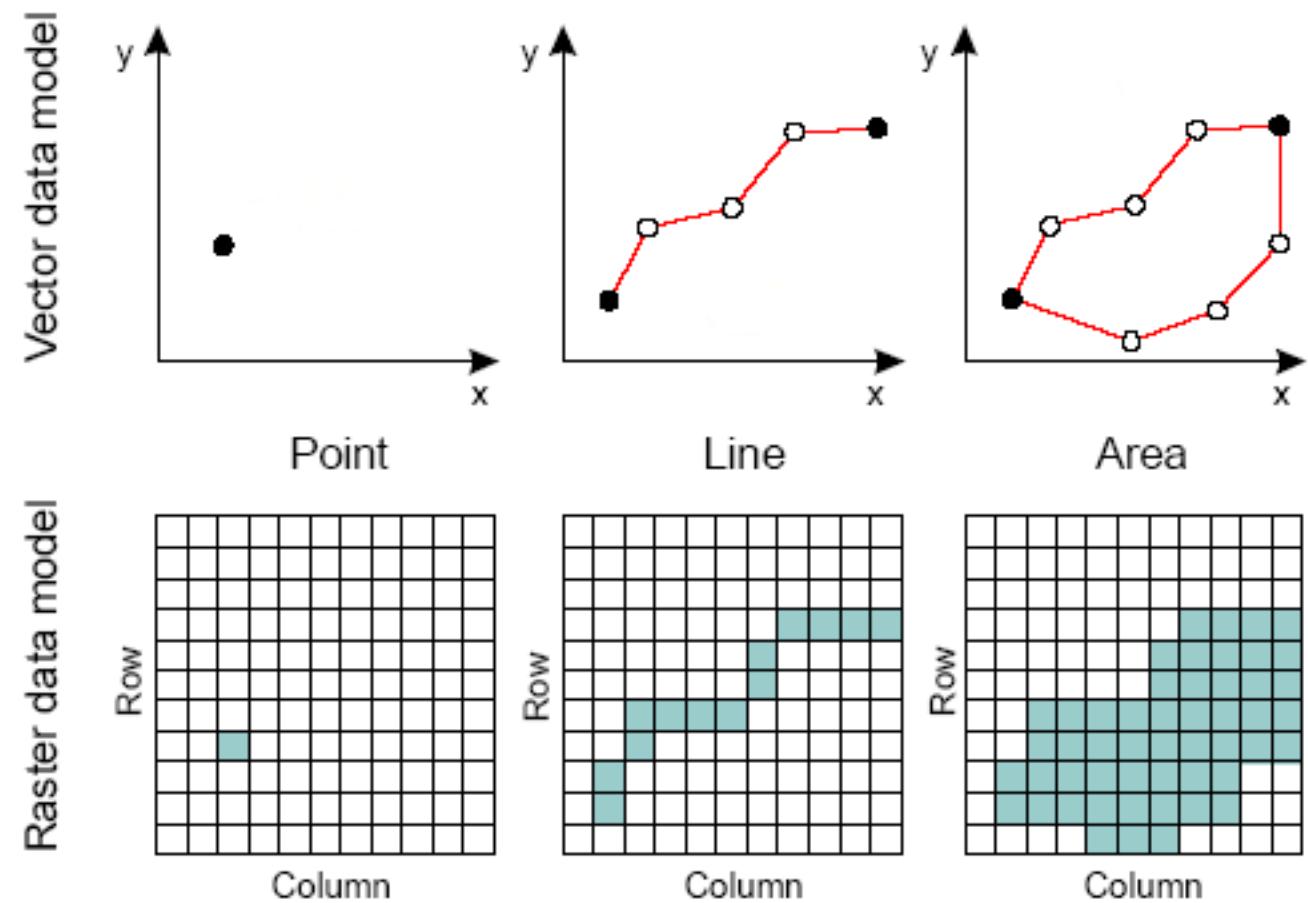
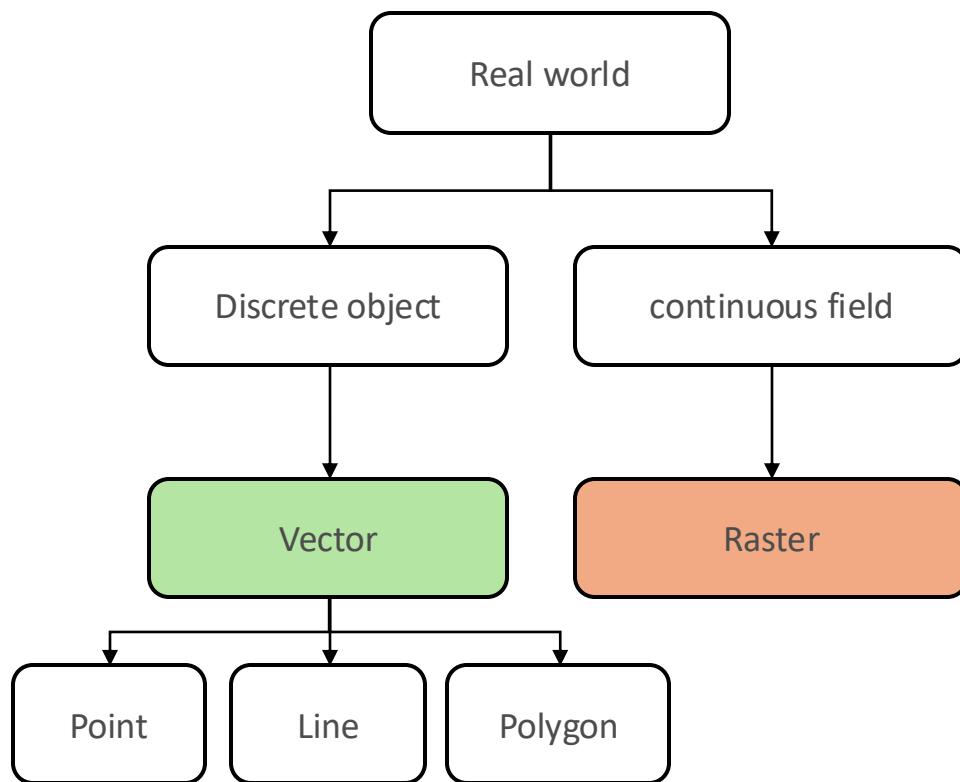
- เป็นการเรียกใช้ method หลายตัวต่อเนื่องจาก object เดียว
- ลดการสร้างตัวแปรชั่วคราวที่ไม่จำเป็น
- ช่วยแสดงลำดับของกระบวนการที่ได้ชัดเจน เช่น preprocessing -> calculation -> renaming

```
var ndvi = ee.Image('COPERNICUS/S2/20240101T000000')
  .clip(ee.Geometry.Point(100.5, 13.7).buffer(5000))
  .normalizedDifference(['B8', 'B4'])
  .rename('NDVI');
```

ประเภทข้อมูลหลักใน GEE

- ee.Geometry รูปทรงเชิงพื้นที่ เช่น จุด เส้น พื้นที่
- ee.Feature ข้อมูลเชิงพื้นที่พร้อมคุณสมบัติ (Geometry + Attribute)
- ee.FeatureCollection กลุ่มของ feature
- ee.Image ข้อมูลภาพถ่ายจากดาวเทียม เช่น Landsat, Sentinel
- ee.ImageCollection กลุ่มของข้อมูลภาพถ่าย เช่น รายเดือน รายปี

Spatial data transformation



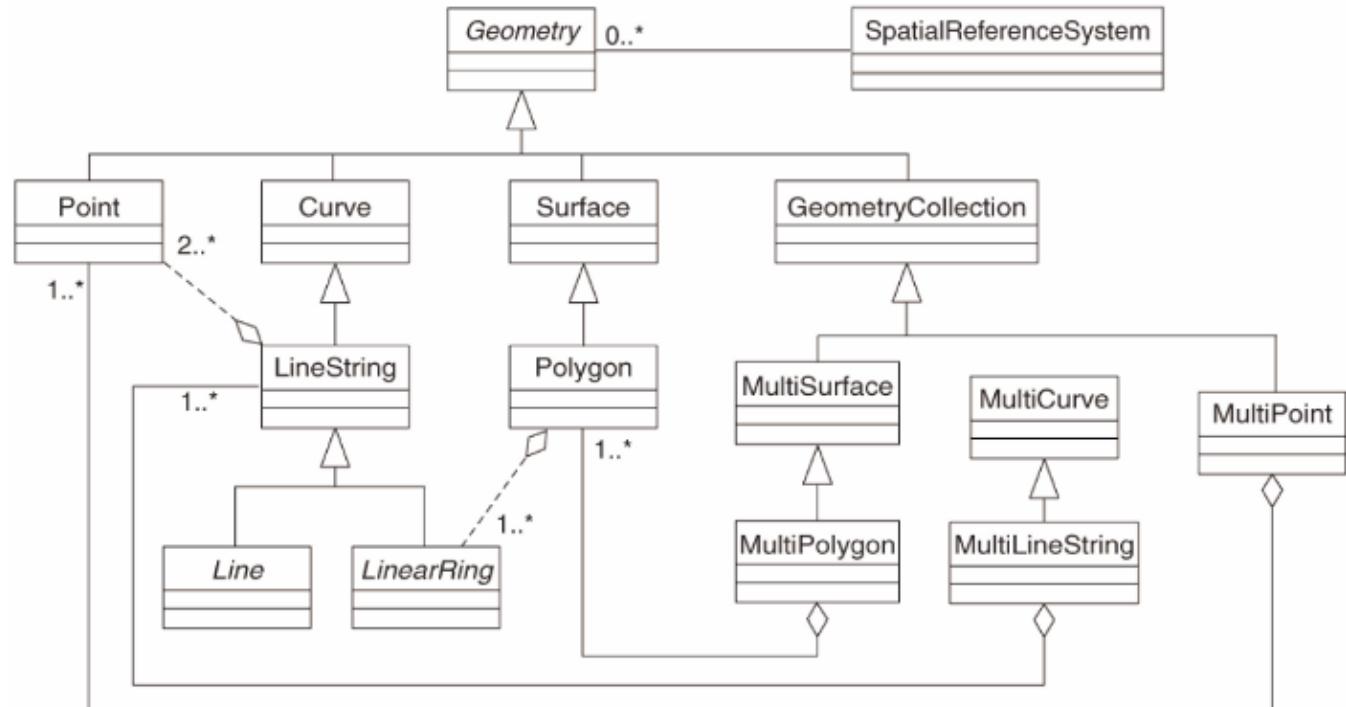
ee.Geometry, ee.Feature, และ ee.FeatureCollection

OGC SFA (Simple Feature Access)

เป็นมาตรฐานจาก **OGC (Open Geospatial Consortium)** สำหรับกำหนดรูปแบบ ข้อมูล เวกเตอร์ ที่ใช้ในระบบ GIS ได้แก่

- Geometry (จุด เส้น พื้นที่)
- Feature (วัตถุเชิงพื้นที่)
- FeatureCollection (กลุ่มของ Feature)

GEE รองรับแนวคิดนี้ ทั้งหมด เช่น
ee.Geometry, ee.Feature, และ
ee.FeatureCollection

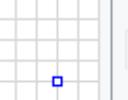
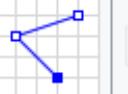
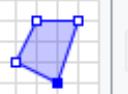
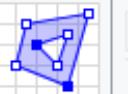


https://docs.qgis.org/3.40/en/docs/training_manual/spatial_database/s/simple_feature_model.html

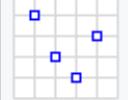
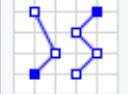
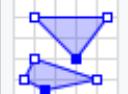
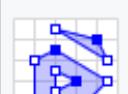
ee.Geometry

- เป็นองค์ประกอบเชิงพื้นที่พื้นฐาน เช่น จุด เส้น หรือพื้นที่ ซึ่งเป็นไปตาม OGC Geometry Types

Geometry primitives (2D)

Type	Examples	
Point		POINT (30 10)
LineString		LINESTRING (30 10, 10 30, 40 40)
Polygon		POLYGON ((30 10, 40 40, 20 40, 10 20, 30 10))
		POLYGON ((35 10, 45 45, 15 40, 10 20, 35 10), (20 30, 35 35, 30 20, 20 30))

Multipart geometries (2D)

Type	Examples	
MultiPoint		MULTIPOINT ((10 40), (40 30), (20 20), (30 10)) MULTIPOINT (10 40, 40 30, 20 20, 30 10)
MultiLineString		MULTILINESTRING ((10 10, 20 20, 10 40), (40 40, 30 30, 40 20, 30 10))
MultiPolygon		MULTIPOLYGON (((30 20, 45 40, 10 40, 30 20), ((15 5, 40 10, 10 20, 5 10, 15 5))))
		MULTIPOLYGON (((40 40, 20 45, 45 30, 40 40), (20 35, 10 30, 10 10, 30 5, 45 20, 20 35), (30 20, 20 15, 20 25, 30 20)))
GeometryCollection		GEOMETRYCOLLECTION (POINT (40 10), LINESTRING (10 10, 20 20, 10 40), POLYGON ((40 40, 20 45, 45 30, 40 40)))

- ตัวอย่าง

```
var point = ee.Geometry.Point([98.9171009716561, 18.815619476862654]);
var line = ee.Geometry.LineString([[98.9171009716561, 18.815619476862654], [99.0873890575936, 18.68557890893041]]);
var polygon = ee.Geometry.Polygon(
  [[[98.9171009716561, 18.815619476862654],
    [98.9171009716561, 18.68557890893041],
    [99.0873890575936, 18.68557890893041],
    [99.0873890575936, 18.815619476862654]]]);
```

ee.Feature

เป็นข้อมูลที่ประกอบด้วย

- **Geometry** (เป็นข้อมูลเชิงพื้นที่ เช่น point, line, polygon)
- **Properties** (เป็นข้อมูลไม่เชิงพื้นที่ เช่น name, id)

Feature: JSON
 Feature (Point, 2 properties) JSON
 type: Feature
 geometry: Point (98.92, 18.82)
 properties: Object (2 properties)

Feature: JSON
 Feature (Polygon, 2 properties) JSON
 type: Feature
 geometry: Polygon, 5 vertices
 properties: Object (2 properties)

- ตัวอย่าง

```
// Feature object
var point = ee.Geometry.Point([98.9171009716561, 18.815619476862654]);
var feature = ee.Feature(point, {name: 'Chiang Mai', population: 1000000});
print('Feature:', feature);

// Polygon feature
var polygon = ee.Geometry.Polygon(
  [[[98.9171009716561, 18.815619476862654],
    [98.9171009716561, 18.68557890893041],
    [99.0873890575936, 18.68557890893041],
    [99.0873890575936, 18.815619476862654]]]);
var feature = ee.Feature(polygon, {name: 'Chiang Mai', population: 1000000});
print('Feature:', feature);
// add to map
Map.centerObject(feature, 10);
Map.addLayer(feature, {color: 'red'}, 'Feature');
```

ee.FeatureCollection

- เป็น ชุดของ Feature

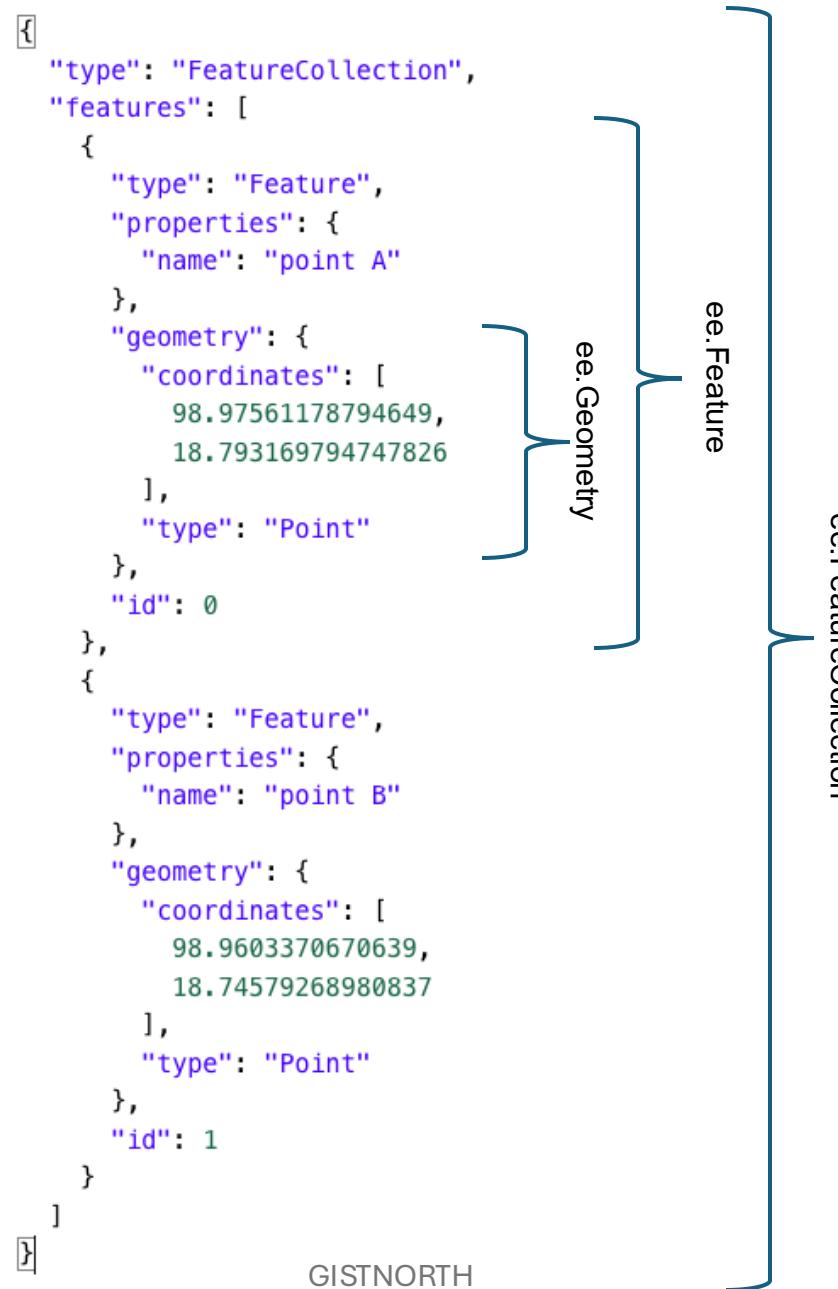
```
‐ FeatureCollection (2 elements, 3 columns)      JSON
    type: FeatureCollection
    ‐ columns: Object (3 properties)
    ‐ features: List (2 elements)
        ‐ 0: Feature 0 (Point, 2 properties)
            type: Feature
            id: 0
            ‐ geometry: Point (98.92, 18.82)
            ‐ properties: Object (2 properties)
        ‐ 1: Feature 1 (Point, 2 properties)
            type: Feature
            id: 1
            ‐ geometry: Point (99.09, 18.69)
            ‐ properties: Object (2 properties)
```

- ตัวอย่าง

```
var point1 = ee.Geometry.Point([98.9171009716561, 18.815619476862654]);
var point2 = ee.Geometry.Point([99.0873890575936, 18.68557890893041]);
var feature1 = ee.Feature(point1, {name: 'Chiang Mai', population: 1000000});
var feature2 = ee.Feature(point2, {name: 'Sarapee', population: 8000000});
var featureCollection = ee.FeatureCollection([feature1, feature2]);
print('Feature Collection:', featureCollection);
// add to map
Map.centerObject(featureCollection, 10);
Map.addLayer(featureCollection, {color: 'red'}, 'Feature Collection');
```

geojson

```
{  
  "type": "FeatureCollection",  
  "features": [  
    {  
      "type": "Feature",  
      "properties": {  
        "name": "point A"  
      },  
      "geometry": {  
        "coordinates": [  
          98.97561178794649,  
          18.793169794747826  
        ],  
        "type": "Point"  
      },  
      "id": 0  
    },  
    {  
      "type": "Feature",  
      "properties": {  
        "name": "point B"  
      },  
      "geometry": {  
        "coordinates": [  
          98.9603370670639,  
          18.74579268980837  
        ],  
        "type": "Point"  
      },  
      "id": 1  
    }  
  ]  
}
```

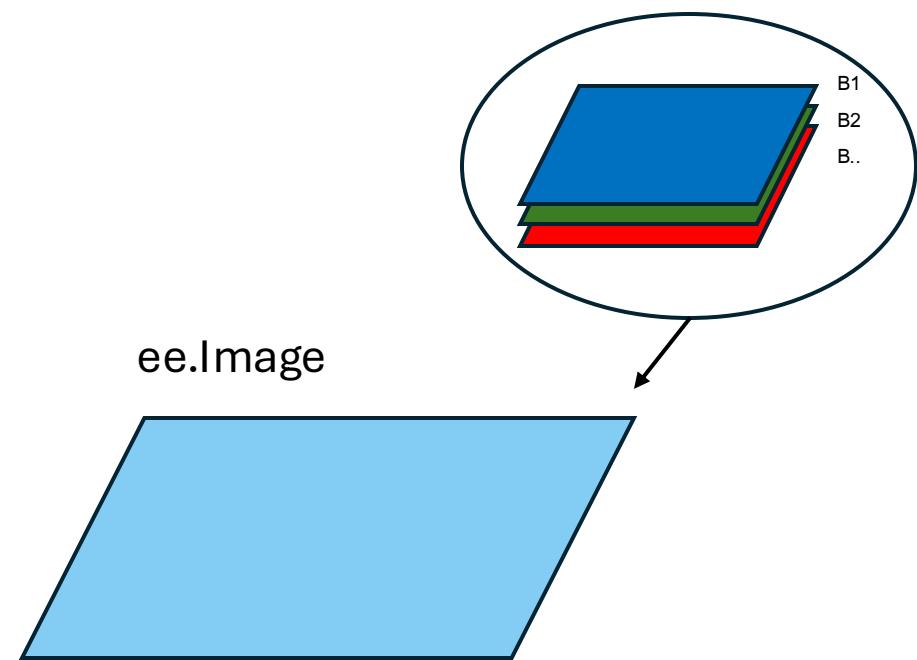


GISTNORTH

ee.Image

ee.Image: ภาพเดี่ยว (Single Image)

- คือภาพ (Raster Image) ที่ประกอบด้วยข้อมูลแบบหลาย band ซึ่งแต่ละ band แทนข้อมูลจากคลื่นแม่เหล็กไฟฟ้าต่าง ๆ เช่น Red, Green, NIR เป็นต้น
- ใช้สำหรับการวิเคราะห์ข้อมูลแบบพิกเซล เช่น คำนวณ NDVI, การแยกแหล่งที่ดิน, การคำนวณดัชนีต่าง ๆ
- สามารถใช้ method เช่น .select(), .clip(), .addBands(), .normalizedDifference() กับ ee.Image



ตัวอย่าง Method สำหรับ ee.Image

เมธอด	คำอธิบาย	ตัวอย่าง
select()	เลือกแบนด์ (band) ที่ต้องการจากภาพ	image.select('B4')
addBands()	เพิ่มแบนด์ใหม่เข้าไปในภาพ	image.addBands(ndvi)
multiply(), divide(), add(), subtract()	คำนวณค่าพิกเซลแต่ละจุด (pixel-wise operation)	image.multiply(0.0001)
normalizedDifference()	คำนวณดัชนี normalized เช่น NDVI	image.normalizedDifference(['B5', 'B4'])
clip()	ตัดภาพตามขอบเขตที่กำหนด	image.clip(geometry)
mask()	กำหนดเพื่อแสดงเฉพาะพิกเซลที่ต้องการ	image.mask(cloudMask)
updateMask()	อัปเดต mask เดิม โดยซ้อนกับ mask ใหม่	image.updateMask(waterMask)
reduceRegion()	คำนวณค่าทางสถิติ เช่น mean, sum ภายใน geometry	image.reduceRegion({reducer: ee.Reducer.mean(), geometry: roi})
reduce()	ใช้กับ reducer เช่น mean, min, max สำหรับทุกแบนด์	image.reduce(ee.Reducer.mean())
rename()	เปลี่ยนชื่อแบนด์	image.rename(['Red', 'Green', 'Blue'])
resample()	ปรับวิธีการ resampling (เช่น bilinear, nearest)	image.resample('bilinear')
reproject()	กำหนดระบบพิกัดใหม่และความละเอียด	image.reproject('EPSG:4326', null, 30)
reduceConnectedComponents()	วิเคราะห์กลุ่มของพิกเซลติดกัน	ใช้ในการจำแนกวัตถุ
visualize()	เตรียมภาพให้แสดงผลได้ เช่น RGB	image.visualize({bands: ['B4','B3','B2'], min: 0, max: 3000})

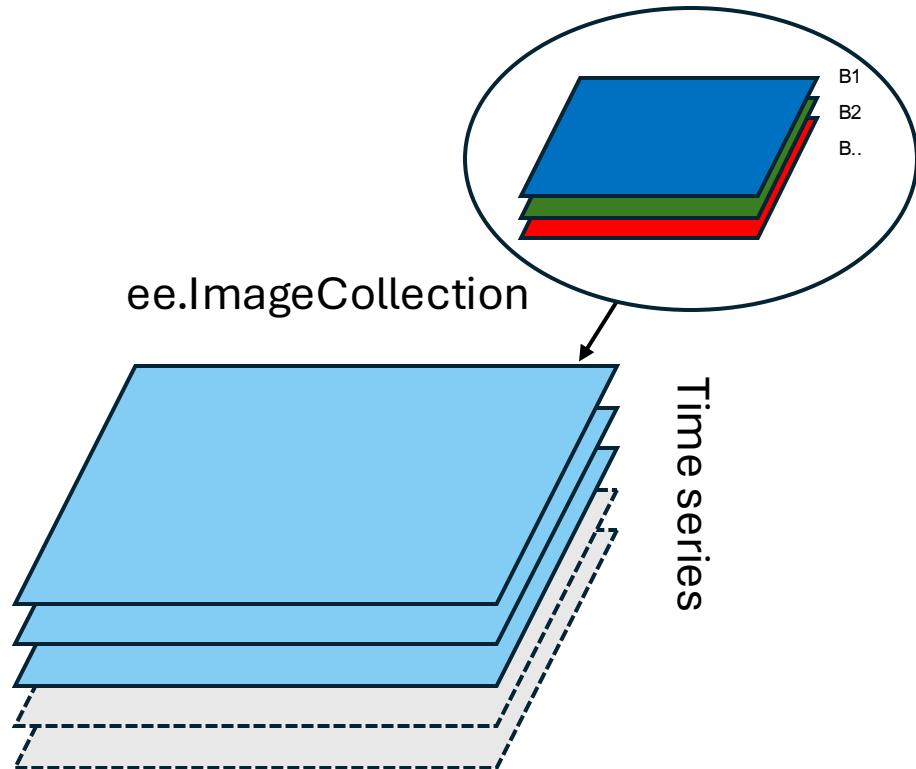
- ตัวอย่าง

```
// Image object
var image = ee.Image('LANDSAT/LC09/C02/T1_L2/LC09_129050_20231220')
    .multiply(0.0000275);
var band4 = image.select('SR_B4');
var band3 = image.select('SR_B3');
var band2 = image.select('SR_B2');
var rgb = band4.addBands(band3).addBands(band2);
Map.centerObject(image, 10);
Map.addLayer(rgb, {min: 0.2, max: 0.6, gamma: 2.0}, 'RGB');
```

ee.ImageCollection

ee.ImageCollection: กลุ่มภาพหลายภาพ (Image Set) ที่สามารถจัดการวิเคราะห์แบบชุดข้อมูลได้

- คือกลุ่มของ ee.Image หลาย ๆ ภาพที่จัดอยู่ในลักษณะเป็นคอลเลกชัน เช่น ภาพรายวัน รายเดือน รายปี หรือทุกภาพจากพื้นที่หนึ่ง
- ใช้สำหรับการวิเคราะห์ภาพในช่วงเวลาหรือหลายช่วงข้อมูล เช่น median, max, mean, หรือคำนวณแนวโน้ม (trend)
- มี method ที่สำคัญ เช่น .filterDate(), .filterBounds(), .map(), .median(), .mean()



- ตัวอย่าง

```
var geometry = ee.Geometry.Polygon(  
  [[[98.9171009716561, 18.815619476862654],  
   [98.9171009716561, 18.68557890893041],  
   [99.0873890575936, 18.68557890893041],  
   [99.0873890575936, 18.815619476862654]]]);  
  
var collection = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED')  
  .filterDate('2021-01-01', '2021-01-31')  
  .filterBounds(geometry)  
  .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 20))  
  .select(['B4', 'B3', 'B2'])  
  .median()  
  .multiply(0.0001);  
  
Map.centerObject(geometry, 10);  
Map.addLayer(collection, {bands: ['B4', 'B3', 'B2'], min: 0.0, max: 0.3, gamma: 1.5}, 'Median Image')
```

ตัวอย่าง Method สำหรับ ee.ImageCollection

เมธอด	คำอธิบาย	ตัวอย่างการใช้งาน
filterDate(start, end)	กรองภาพตามช่วงวันที่	.filterDate('2023-01-01', '2023-12-31')
filterBounds(geometry)	กรองภาพตามขอบเขตพื้นที่	.filterBounds(roi)
filterMetadata()	กรองภาพจากค่าคุณสมบัติ (metadata) เช่น CLOUD_COVER	.filterMetadata('CLOUD_COVER', 'less_than', 10)
select(bands)	เลือกเฉพาะบางแบนด์	.select(['B4', 'B3', 'B2'])
map(function)	ใช้ฟังก์ชันกับทุกภาพใน collection	.map(function(img){ return img.normalizedDifference(['B8','B4']).rename('NDVI'); })
mean(), median(), min(), max(), sum(), stdDev()	คำนวณค่าสถิติของแต่ละพิกเซลข้าม collection	.median()
mosaic()	นำภาพทั้งหมดมาทับซ้อนกัน โดยใช้ค่าจากภาพที่อยู่บนสุด	.mosaic()
first(), limit(n)	เลือกภาพแรก, จำกัดจำนวนภาพ	.first() หรือ .limit(10)
sort(property, ascending)	เรียงลำดับภาพตามค่า metadata เช่น CLOUD_COVER	.sort('CLOUD_COVER')
size()	คืนจำนวนภาพใน collection	.size()
toList(n)	แปลงเป็น list ของภาพ	.toList(5)

Reduce

- **Reduce** คือการสรุปค่าข้อมูลจากหลาย ๆ พิกเซล หลายช่วงเวลา หรือหลาย band ให้เป็นค่าที่เรียบง่าย ขึ้น เช่น ค่าเฉลี่ย ค่าสูงสุด ค่าต่ำสุด หรือค่ามัธยฐาน
- ประเภทของ Reducers

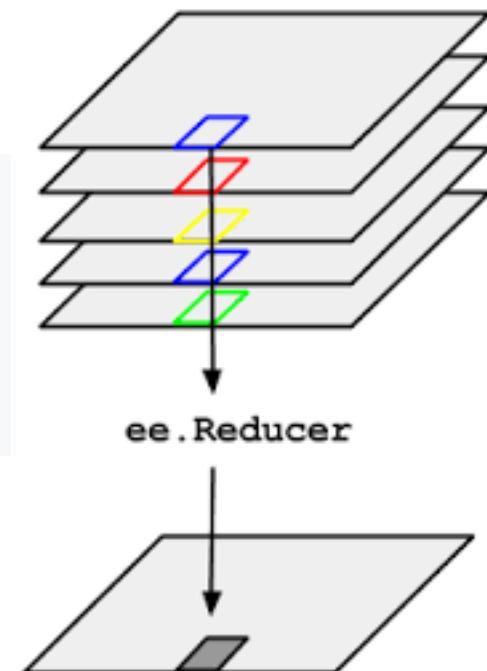
ประเภท	ใช้กับ	เป้าหมาย	ตัวอย่างเมธอด
Time Reducer	ImageCollection	รวมค่าตามเวลา	.reduce(ee.Reducer.mean())
Spatial Reducer	Image	รวมค่าตามพื้นที่ (geometry)	.reduceRegion(), .reduceRegions()
Band Reducer	Image หลาย band	รวมค่าระหว่าง band	.reduce(ee.Reducer.mean())

1. Time Reducers (การย่อข้อมูลตามเวลา)

- ใช้กับ ImageCollection เพื่อย่อหlaysภาพในช่วงเวลาหนึ่งให้เหลือเพียงภาพเดียว เช่น ใช้ mean(), median(), max(), min() เพื่อหาค่าเฉลี่ยรายปี/เดือน/วัน เป็นต้น

```
var s2 = ee.ImageCollection('COPERNICUS/S2')
    .filterDate('2021-01-01', '2021-12-31')
    .filterBounds(polygon);

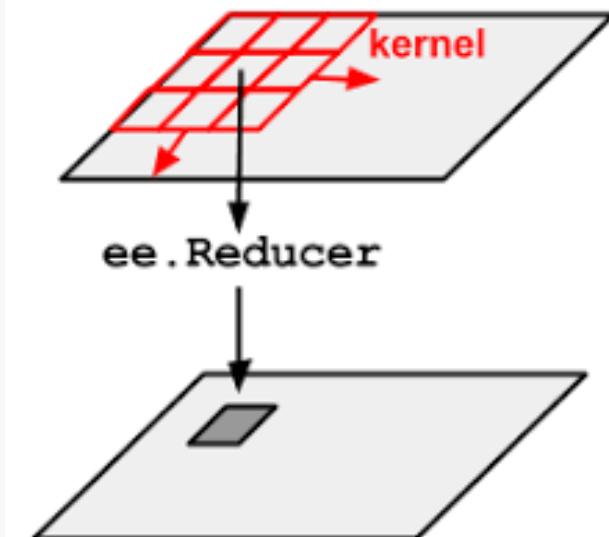
// Compute per-pixel mean across time
var meanTime = s2.reduce(ee.Reducer.mean());
```



2. Spatial Reducers (การย่อข้อมูลเชิงพื้นที่)

- ใช้ลดค่าพิกเซลภายในพื้นที่กำหนด เช่น พื้นที่ขอบเขตตำบล หรือ buffer
- ทำงานร่วมกับ .reduceRegion() หรือ .reduceRegions()
- เหมาะสมสำหรับการคำนวณค่าเฉลี่ย NDVI, LST, หรือข้อมูลอื่นๆ ภายใน polygon

```
var polygon = ee.Geometry.Polygon(  
  [[[98.9171009716561, 18.815619476862654],  
   [98.9171009716561, 18.68557890893041],  
   [99.0873890575936, 18.68557890893041],  
   [99.0873890575936, 18.815619476862654]]]);  
  
var s2 = ee.ImageCollection('COPERNICUS/S2')  
    .filterDate('2021-01-01','2021-03-31')  
    .filterBounds(polygon);  
  
var composite = s2.median();  
var bandStats = composite.reduceRegion({  
  reducer: ee.Reducer.mean(),  
  geometry: polygon,  
  scale: 100  
});
```



3. Band Reducers (การย่อข้อมูลระหว่าง band)

- ใช้กับ ee.Image ที่มีหลาย band และต้องการสรุปค่าจากหลาย band เป็นค่ารวมเดียว เช่น คำนวณค่าเฉลี่ยของ band ทุกอัน
- ได้ค่าผลรวมจากหลาย band เช่น ค่าความเข้มเฉลี่ยของ RGB หรือ spectral index หลายตัวรวมกัน

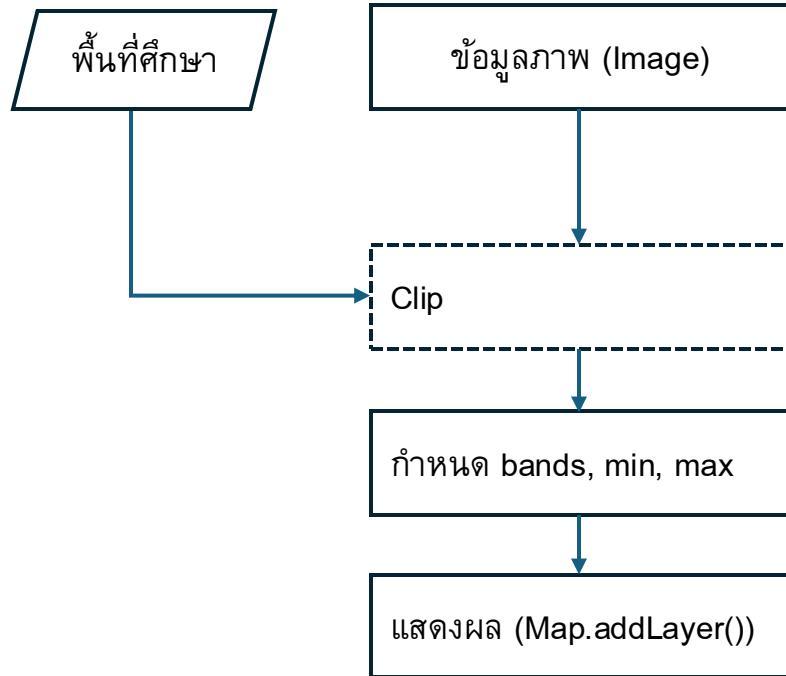
```
var s2 = ee.ImageCollection('COPERNICUS/S2')
    .filterDate('2021-01-01', '2021-12-31')
    .filterBounds(polygon)
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 20))
    .select(['B8', 'B4']);

var ndviCollection = s2.map(function(img) {
  var ndvi = img.normalizedDifference(['B8', 'B4']).rename('NDVI');
  return ndvi.copyProperties(img, img.propertyNames());
});

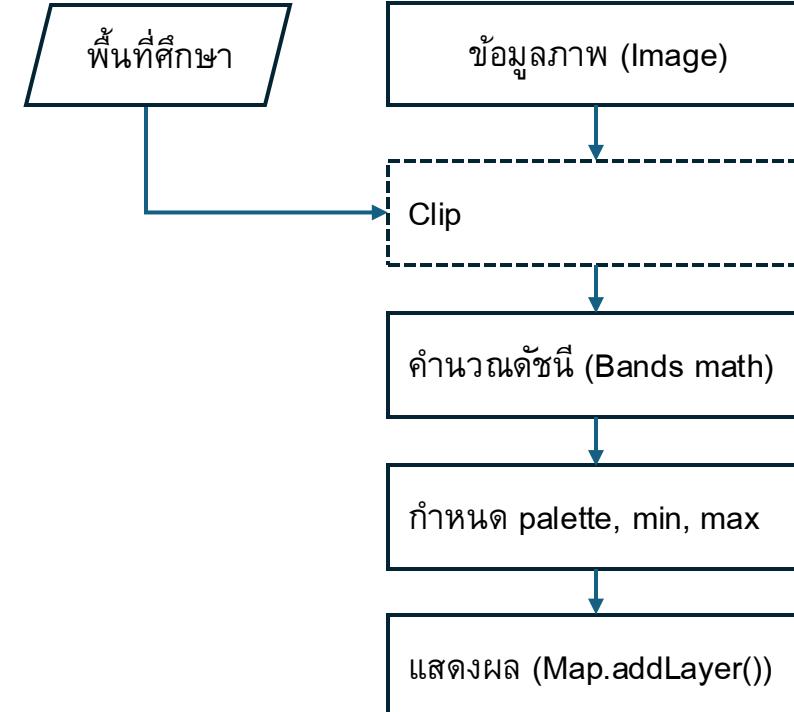
var meanNDVI = ndviCollection.reduce(ee.Reducer.mean());
```

การทำงานกับข้อมูล Image

Band composite



แสดงผลจากการวิเคราะห์ spectral indices



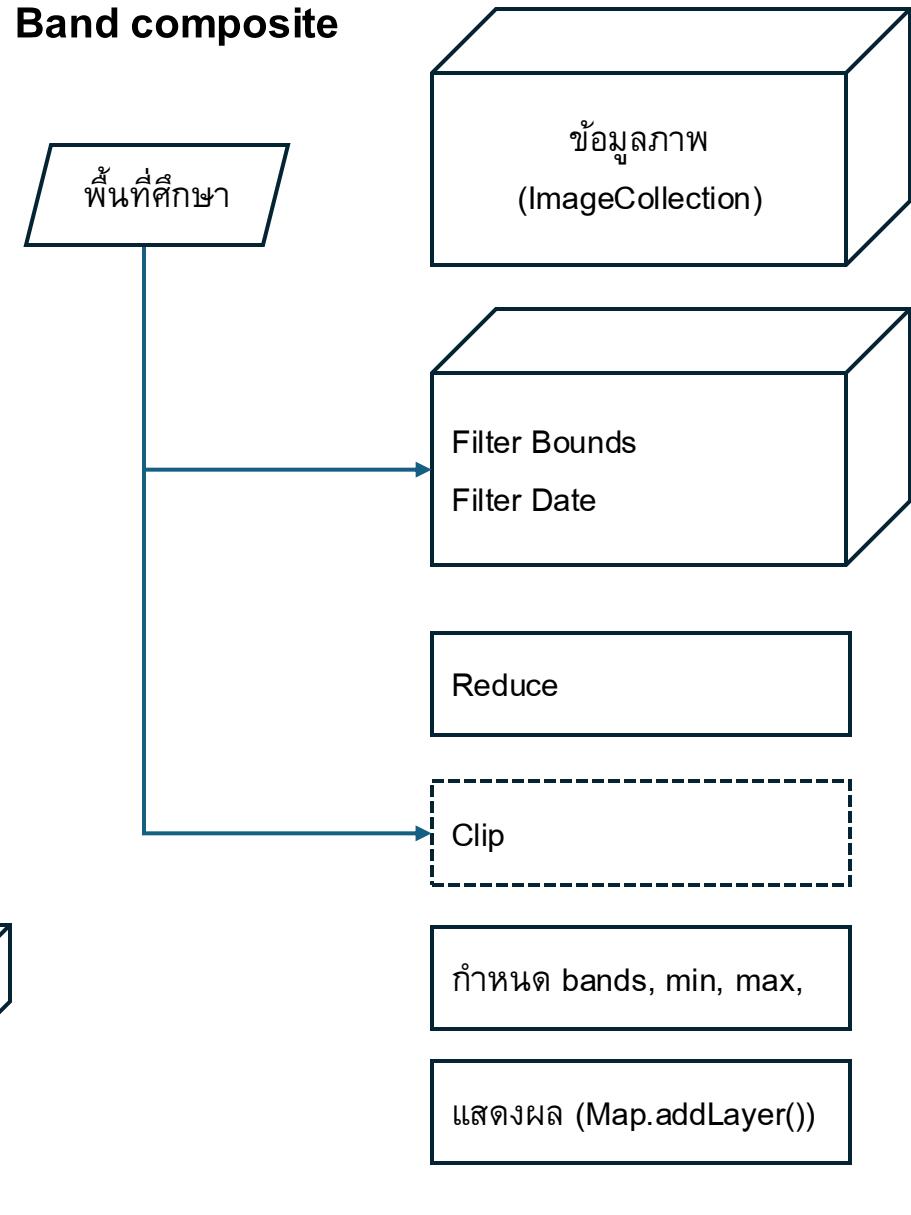
ImageCollection

Image

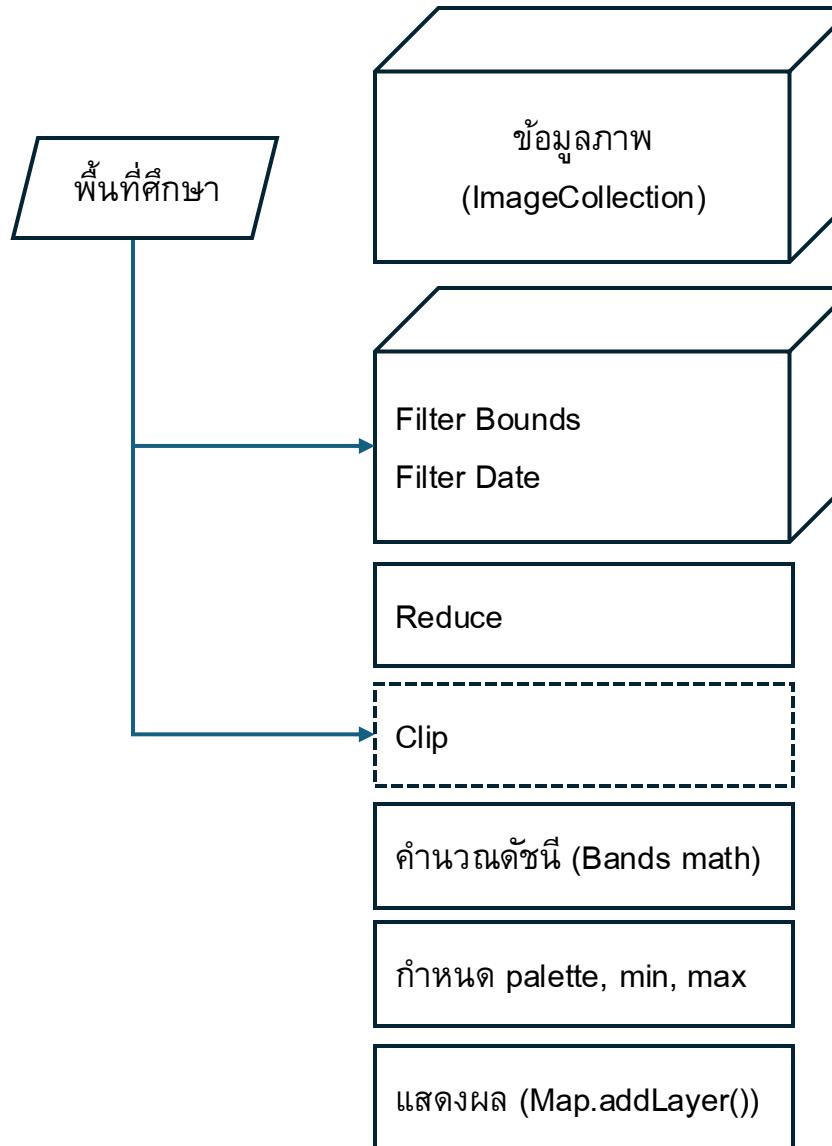
vector

การทำงานกับข้อมูล ImageCollection

Band composite



วิเคราะห์ spectral indices



การแสดงผลภาพ (Map.addLayer())

```
Map.addLayer(image, visParams, 'ชื่อชั้นข้อมูล');
```

- image: ภาพหรือข้อมูลที่จะนำมาแสดง (เช่น NDVI, Sentinel-2)
- visParams: การตั้งค่าการแสดงผล เช่น band, min/max, palette
- ชื่อชั้นข้อมูล: ชื่อที่แสดงใน Layers Panel
- ตัวอย่าง

```
var vis = {bands: ['B4', 'B3', 'B2'], min: 0, max: 3000};  
Map.addLayer(image, vis, 'ภาพ RGB');
```

การแสดงผล: จากการเลือก Band

- การเลือก Band ที่ใช้ขึ้นกับวัตถุประสงค์ เช่น

ประเภทภาพ	ดาวเทียม	Band ที่ใช้	คำอธิบาย
True Color	Landsat 8/9	B4 (Red), B3 (Green), B2 (Blue)	เหมือนตามการมองเห็นของตาบุคคล
True Color	Sentinel-2	B4 (Red), B3 (Green), B2 (Blue)	ใช้ดูภูมิประเทศทั่วไป, เมือง, น้ำ
False Color (Vegetation)	Landsat 8/9	B5 (NIR), B4 (Red), B3 (Green)	พืชพรรณจะแสดงเป็นสีเขียวเข้ม (สุขภาพดี)
False Color (Vegetation)	Sentinel-2	B8 (NIR), B4 (Red), B3 (Green)	ใช้ตรวจสอบสุขภาพพืช และพื้นที่เกษตร
False Color (SWIR)	Landsat 8/9	B7 (SWIR2), B5 (NIR), B4 (Red)	เน้นความชื้น ดิน ป่าไม้ ไฟป่า
False Color (SWIR)	Sentinel-2	B12 (SWIR2), B8 (NIR), B4 (Red)	ใช้ในการติดตามน้ำท่วม ไฟป่า และดิน

- ตัวอย่างโค้ดการแสดงผลแบบ True Color (Sentinel-2)

```
var image = ee.Image('COPERNICUS/S2_SR/20220101T000000');
Map.addLayer(image, {
  bands: ['B4', 'B3', 'B2'],
  min: 0,
  max: 3000
}, 'True Color');
```

- ตัวอย่างโค้ดการแสดงผลแบบ False Color (Vegetation)

```
Map.addLayer(image, {
  bands: ['B8', 'B4', 'B3'],
  min: 0,
  max: 3000
}, 'False Color Vegetation');
```

การแสดงผล: ด้วย palette สำหรับข้อมูลเชิงค่าต่อเนื่อง (raster)

- พาเลตสีใช้ในการเน้นค่าข้อมูล เช่น NDVI, ความสูง, ความชื้น
- ใช้ร่วมกับคำสั่ง Map.addLayer() โดยกำหนด palette: ['สี1', 'สี2', ...]
- ตัวอย่าง แสดงผล NDVI ด้วยพาเลตสีเขียว

```
var ndviParams = {  
  min: 0,  
  max: 1,  
  palette: ['white', 'green']  
};  
Map.addLayer(ndvi, ndviParams, 'NDVI');
```

- ตัวอย่าง แสดงผลค่าความสูงจาก SRTM

```
var srtm = ee.Image('USGS/SRTMGL1_003');
var visElev = {
  min: 0,
  max: 3000,
  palette: ['blue', 'green', 'yellow', 'orange', 'red']
};
Map.addLayer(srtm, visElev, 'SRTM Elevation');
```

การแสดงผลแบบเวกเตอร์ (Vector Data)

- ตัวอย่าง: แสดงขอบเขตการปกรอง

```
var province = ee.FeatureCollection('TIGER/2018/States');
Map.addLayer(province, {
  color: 'blue'
}, 'ขอบเขตจังหวัด');
```

- ตัวอย่าง: การใส่สีพื้นที่ด้วย fillColor

```
var district = ee.FeatureCollection('FAO/GAUL_SIMPLIFIED_5
Map.addLayer(district.style({
  color: 'black',
  fillColor: 'yellow',
  width: 1
}), {}, 'เขตการปกรอง');
```

การส่งออกข้อมูล

GEE ให้เรา Export ข้อมูลทั้งแบบ Raster (GeoTIFF) และ Vector (SHP) ไปยัง Google

```
Export.image.toDrive({
  image: s2, // ภาพที่ต้องการส่งออก
  description: 'Sentinel2_Median_Jan2021', // ชื่อไฟล์ใน Google Drive
  folder: 'GEE_Exports', // ชื่อโฟลเดอร์ใน Google Drive
  fileNamePrefix: 'S2_Median_Jan2021', // ชื่อไฟล์นำหน้า
  region: roi, // พื้นที่ที่ต้องการส่งออก
  scale: 10, // ความละเอียดของพื้นที่ (เมตร)
  crs: 'EPSG:4326', // ระบบพิกัด
  maxPixels: 1e13 // จำนวนพิกเซลสูงสุดท่อนุญาตให้ส่งออก
});
```

สรุป

GEE เป็นแพลตฟอร์ม วิเคราะห์ข้อมูลภูมิสารสนเทศบนคลาวด์

- มีคลังข้อมูลมากกว่า 80 PB (เช่น Landsat, Sentinel, MODIS)
- ประมวลผลแบบขนาน → เร็ว ไม่ต้องโหลดข้อมูลมาเก็บเอง
- มี API ทั้ง JavaScript, Python และ REST

พื้นฐาน GEE

- ความเข้าใจ Client vs Server side
- การใช้ตัวแปร พิงก์ชัน OOP
- Object หลัก: ee.Geometry, ee.Feature, ee.Image, ee.ImageCollection

การสำรวจจากระยะไกล (Remote Sensing)

- แบ่งเป็น 2 ประเภท: Passive (รับพลังงาน) / Active (ส่งสัญญาณ)
- ความละเอียด 4 มิติ: Spatial, Spectral, Temporal, Radiometric
- ตัวอย่างดาวเทียม: Landsat, Sentinel, MODIS, SAR
- ดัชนีสำคัญ: NDVI, NDWI, NDBI