CE Hub Api Service

Documentation

Contents

[1 Introduction 3](#_Toc161242843)

[2 Access permission 4](#_Toc161242844)

[2.1 API key 4](#_Toc161242845)

[3 Forecast API 4](#_Toc161242846)

[3.1 Authorization 4](#_Toc161242847)

[3.2 Types of forecast data sources 6](#_Toc161242848)

[3.3 Methods 6](#_Toc161242849)

[3.3.1 ShortRangeForecastHourly 6](#_Toc161242850)

[3.3.2 ShortRangeForecastDaily 9](#_Toc161242851)

[3.3.3 Nowcast 11](#_Toc161242852)

[3.4 List of Measure Label 13](#_Toc161242853)

[4 Historical API 14](#_Toc161242854)

[4.1 JSON Query Structure 15](#_Toc161242855)

[4.1.1 Units 15](#_Toc161242856)

[4.1.2 GeoJSON Geometry 15](#_Toc161242857)

[4.1.3 Output Format 16](#_Toc161242858)

[4.1.4 Time Intervals 17](#_Toc161242859)

[4.1.5 Datasets and Variables 17](#_Toc161242860)

[4.1.6 List of Variable JSON 18](#_Toc161242861)

[4.1.7 Useful Links 19](#_Toc161242862)

[5 Soil API 19](#_Toc161242863)

[5.1.1 Interpret USDA Texture Classes Values​ 19](#_Toc161242864)

# Introduction

The objective of **CEHub**, the **Crop Environmental data Hub** is to provide a central access point

* to Syngenta’s plant environmental data associated with time and geographical coordinates.
* to global, quality controlled, homogeneous and therefore reusable data sets.

CEHub aims to support any Syngenta department with environmental data supply

* to perform large analysis over environmental data
* to develop digital internal and customer-facing decisions support solutions

There are two different API’s:

* **Weather forecast**  
  API tailored to get 7 or 14-day forecast automatically with the best spatial resolution and most frequent updates.
* **Historical environmental data**  
  API tailored to get
  + historical weather
  + vegetation health
  + soil characteristic
  + land use
  + topography

# Access permission

## API key

A key is required for each group to execute API calls.

The API keys for forecast and historical will be shared via the onboarding tool.

# Forecast API

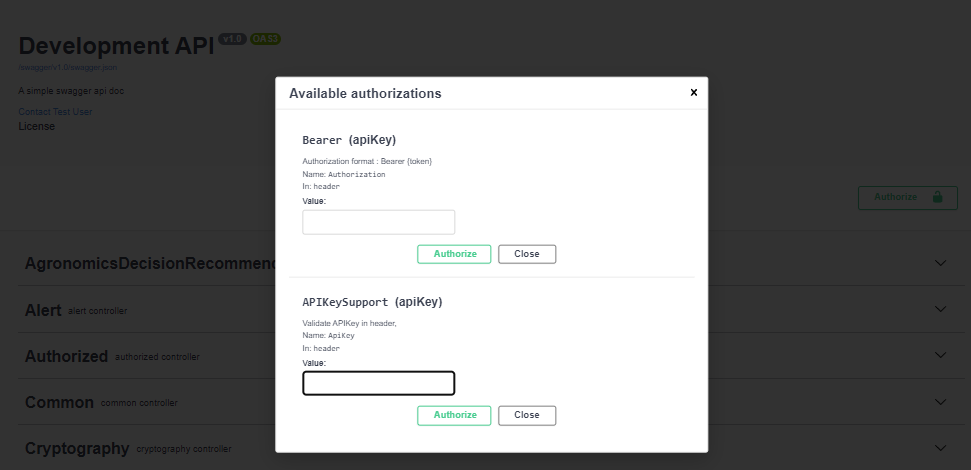
Swagger link : <https://services.cehub.syngenta-ais.com/swagger/index.html>

## Authorization

You can click “Authorize” button on swagger home page.

**A screenshot of a computer

Description automatically generated**



There are 2 ways to Authorized it,

1. Bearer token,
2. Apikey.

kindly use the apikey option which you have obtained from the onboarding tool.

Kindly select the Forecast API’s using the forecast section of the swagger as shown below

A screenshot of a computer

Description automatically generated

## Types of forecast data sources

We work with various third party providers for getting the weather data and the list of providers are mentioned below.

These are the list of providers that we are currently offering.

|  |  |
| --- | --- |
| **providers** | **Data source team** |
| Meteoblue | Meteoblue |
| Iteris | Iteris |
| Climatempo | Climatempo |
| ClimatempoAgro | Climatempo, but different part |

Kindly use **Meteoblue** as the datasource as shown below

A screenshot of a computer

Description automatically generated

## Methods

### ShortRangeForecastHourly

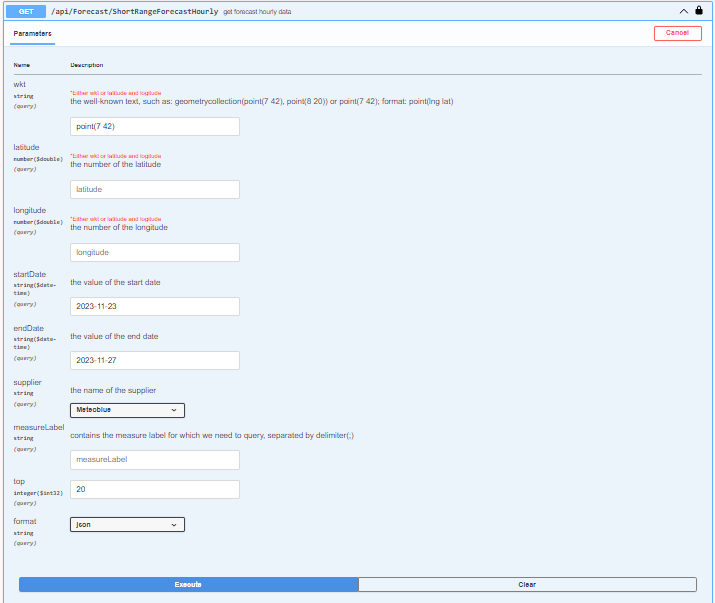
Description

* This method is used to generate the forecast hourly measure data i.e. current conditions of the given location.

* Based on the requested time range, the API provides short range forecast with a time interval of 1 hour
* API will work only with GPS co-ordinates i.e. latitude and longitude or WKT ( explain in section below) and not with zip code or name of the locations.

Input Parameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable Name | Data Type | Required | Description | Default Value | Optional Values |
| wkt | String | No | The well-known text, such as: geometrycollection(point(7 42), point(8 20)) or point(7 42) |  |  |
| latitude | Double | No | The latitude value of the location which want to search |  |  |
| longitude | Double | No | The longitude of the lcation which want to search |  |  |
| startDate | DateTime | No | The sart date of the location which want to search |  |  |
| endDate | DateTime | No | The end date of the location which want to search |  |  |
| supplier | String | No | Only can be one value of optional values | Meteoblue | Meteoblue  Iteris  Climatempo |
| measureLabel | String | No | Contains the measure label for which we need to query, separated by delimiter(;) |  | List of measure Label can be found in the section 3.4 |
| top | Int | No | It will take all data if top is null, otherwise, take top size data | 20 |  |
| format | String | No | Only can be one value of optional values | json | json  csv |



Sample Url

<http://services.cehub.syngenta-ais.com/api/Forecast/ShortRangeForecastHourly?wkt=point%287%2042%29&startDate=2023-11-23&endDate=2023-11-27&supplier=Meteoblue&top=20&format=json&ApiKey=xxxx>

\***kindly change the start date and end date accordingly**

Sample Response



### ShortRangeForecastDaily

* This method is used to generate the forecast daily measure data i.e. current conditions of the given location.

* Based on the requested time range, the API provides short range forecast with a time interval of 1 day
* API will work only with GPS co-ordinates i.e. latitude and longitude or WKT ( explain in section below) and not with zip code or name of the locations.

Input Parameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable Name | Data Type | Required | Description | Default Value | Optional Values |
| wkt | String | No | The well-known text, such as: geometrycollection(point(7 42), point(8 20)) or point(7 42) |  |  |
| latitude | Double | No | The latitude value of the location which want to search |  |  |
| longitude | String | No | The longitude of the lcation which want to search |  |  |
| startDate | DateTime | No | The sart date of the location which want to search |  |  |
| endDate | DateTime | No | The end date of the location which want to search |  |  |
| supplier | String | No | Only can be one value of optional values | Meteoblue | Meteoblue  Iteris  Climatempo  ClimatempoAgro |
| measureLabel | String | No | Contains the measure label for which we need to query, separated by delimiter(;) |  | List of measure Label can be found in the section 3.4 |
| top | Int | No | It will take all data if top is null, otherwise, take top size data | 20 |  |
| format | String | No | Only can be one value of optional values | json | json  csv |

A screenshot of a computer

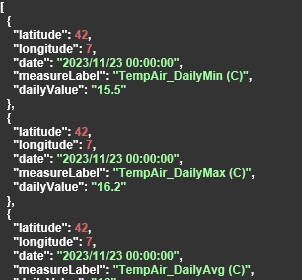
Description automatically generated

Sample Url

<http://services.cehub.syngenta-ais.com/api/Forecast/ShortRangeForecastDaily?wkt=point%287%2042%29&startDate=2023-11-23&endDate=2023-11-27&supplier=Meteoblue&top=20&format=json&ApiKey=xxxx>

**kindly change the start date and end date accordingly**

Sample Response



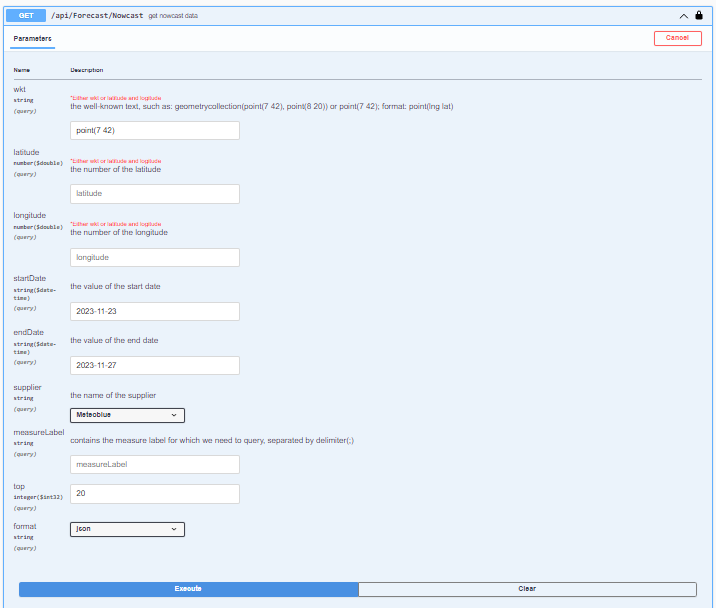
### Nowcast

* This method is used to generate the nowcast i.e. current conditions of the given location.

* Based on the requested time range, the API provides short range forecast with a time interval of 15 mins.
* Currently we support the nowcast data from meteoblue and Iteris.
* API will work only with GPS co-ordinates i.e. latitude and longitude or WKT ( explain in section below) and not with zip code or name of the locations.

Input Parameters

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Variable Name | Data Type | Required | Description | Default Value | Optional Values |
| wkt | String | No | the well-known text, such as: geometrycollection(point(7 42), point(8 20)) or point(7 42); format: point(lng lat)  \*Either wkt or latitude and logitude |  |  |
| latitude | Double | No | the number of the latitude |  |  |
| longitude | String | No | the number of the longitude |  |  |
| startDate | DateTime | No | The start date of the location which want to search |  |  |
| endDate | DateTime | No | The end date of the location which want to search |  |  |
| supplier | String | No | Only can be one value of optional values | Meteoblue | Meteoblue  Iteris |
| measureLabel | String | No | Contains the measure label for which we need to query, separated by delimiter(;) |  | List of measure Label can be found in the section 3.4 |
| top | Int | No | It will take all data if top is null, otherwise, take top size data | 20 |  |
| format | String | No | Only can be one value of optional values | json | json  csv |

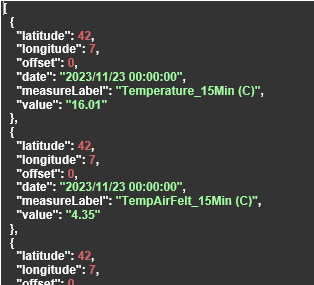


Sample Url

<http://services.cehub.syngenta-ais.com/api/Forecast/Nowcast?wkt=point%287%2042%29&startDate=2023-11-23&endDate=2023-11-27&supplier=Meteoblue&top=20&format=json&ApiKey=xxxx>

**kindly change the start date and end date accordingly**

Sample Response



## List of Measure Label

|  |  |
| --- | --- |
| MeasureType | Label |
| Nowcast | Temperature\_15Min (C) |
| Nowcast | WindSpeed\_15Min (m/s) |
| Nowcast | WindDirection\_15Min |
| Nowcast | HumidityRel\_15Min (pct) |
| Hourly | Cloudcover\_Hourly (pct) |
| Hourly | GlobalRadiation\_HourlySum (Wh/m2) |
| Hourly | HumidityRel\_Hourly (pct) |
| Hourly | Precip\_HourlySum (mm) |
| Hourly | PrecipProbability\_Hourly (pct) |
| Hourly | ShowerProbability\_Hourly (pct) |
| Hourly | SnowFraction\_Hourly |
| Hourly | SunshineDuration\_Hourly (min) |
| Hourly | TempAir\_Hourly (C) |
| Hourly | Visibility\_Hourly (m) |
| Hourly | WindDirection\_Hourly (Deg) |
| Hourly | WindGust\_Hourly (m/s) |
| Hourly | WindSpeed\_Hourly (m/s) |
| Hourly | Soilmoisture\_0to10cm\_Hourly (vol%) |
| Hourly | Soiltemperature\_0to10cm\_Hourly (C) |
| Hourly | Referenceevapotranspiration\_HourlySum (mm) |
| Hourly | LeafWetnessProbability\_Hourly (pct) |
| Daily | Cloudcover\_DailyAvg (pct) |
| Daily | Evapotranspiration\_DailySum (mm) |
| Daily | GlobalRadiation\_DailySum (Wh/m2) |
| Daily | HumidityRel\_DailyAvg (pct) |
| Daily | HumidityRel\_DailyMax (pct) |
| Daily | HumidityRel\_DailyMin (pct) |
| Daily | Precip\_DailySum (mm) |
| Daily | PrecipProbability\_Daily (pct) |
| Daily | ShowerProbability\_DailyMax (pct) |
| Daily | SnowFraction\_Daily (pct) |
| Daily | SunshineDuration\_DailySum (min) |
| Daily | TempAir\_DailyAvg (C) |
| Daily | TempAir\_DailyMax (C) |
| Daily | TempAir\_DailyMin (C) |
| Daily | ThunderstormProbability\_DailyMax (pct) |
| Daily | WindDirection\_DailyAvg (Deg) |
| Daily | WindGust\_DailyMax (m/s) |
| Daily | WindSpeed\_DailyAvg (m/s) |
| Daily | WindSpeed\_DailyMax (m/s) |
| Daily | WindSpeed\_DailyMin (m/s) |
| Daily | WindDirection\_DailyAvg |
| Daily | Soilmoisture\_0to10cm\_DailyMax (vol%) |
| Daily | Soilmoisture\_0to10cm\_DailyAvg (vol%) |
| Daily | Soilmoisture\_0to10cm\_DailyMin (vol%) |
| Daily | Soiltemperature\_0to10cm\_DailyMax (C) |
| Daily | Soiltemperature\_0to10cm\_DailyAvg (C) |
| Daily | Soiltemperature\_0to10cm\_DailyMin (C) |
| Daily | Referenceevapotranspiration\_DailySum (mm) |

# Historical API

The Historical API grants access to the entire weather data archive, which is comprised of various weather variables gathered from over 50 data sources.

URL : <http://my.meteoblue.com/dataset/query?apikey=APIKEY>

Kindly replace APIKEY value with the key obtained from the onboarding tool.

We cannot use the GET parameters is not sufficient to query datasets dynamically. Instead a HTTP JSON request body is used:

{

"units": {

"temperature": "C",

"velocity": "km/h",

"length": "metric",

"energy": "watts"

},

"geometry": {

"type": "MultiPoint",

"coordinates": [[7.57327,47.558399,279]], // lon, lat, asl

"locationNames": ["Basel"]

},

"format": "json",

"timeIntervals": [

"2019-01-01T+00:00/2019-12-31T+00:00"

],

"queries": [{

"domain": "NEMSGLOBAL",

"gapFillDomain": null,

"timeResolution": "hourly",

"codes": [{

"code": 157,

"level": "180-0 mb above gnd"

}]

}]

}

## JSON Query Structure

The JSON body uses various structures and arrays that are nested to build complex queries with recursive transformations. All JSON attributes are case-sensitive and use camel-case names. As in the example above, the outer JSON structure contains properties like units, geometry, timeIntervals or queries*.*

| **Property** | **Type** | **Description** |
| --- | --- | --- |
| units | Structure: Units | Option to select units like Fahrenheit |
| geometry | Structure: GeoJSON | Select polygon or points |
| format | String enumeration: Format | Which output format to use |
| timeIntervals | Array of Structure: TimeInterval | Define time intervals to read |
| queries | Array of Structure: Query | Per dataset queries |

### Units

If units are not set, the defaults are Celsius, km/h, metric and watts

| **Property** | **Type** | **Description** |
| --- | --- | --- |
| temperature | String | c(Celsius) or f (Fahrenheit) |
| velocity | String | km/h, m/s , mph, kn or bft |
| length | String | metric or imperial |
| energy | String | watts or joules |

### GeoJSON Geometry

**Please make sure to provide all input coordinates in the correct order: "lon" -> "lat" (-> "asl")**

The geometry structure is based on GeoJSON, but extended to support features like location names and additional attributes.

**Coordinates are defined as tuple of longitude, latitude and elevation above sea level.** Elevation is optional and will be automatically resolved from an 80 m resolution digital elevation model (DEM). locationNames can be optionally specified and will be replicated in the output.

For Polygon and multiPolygon,The first and last coordinate must be the same. Please make sure to supply a valid polygon without self-intersections.

The sample GeoJSON body for

Point

{

"type": "Point",

"coordinates": [8.6, 47.5, 351.1] // lon, lat, asl

}

MultiPoint

{

"type": "MultiPoint",

"coordinates": [[8.6, 47.5,351.1], [8.55, 47.37, 429]], // lon, lat, asl

"locationNames": ["Basel", "Zürich"]

}

Polygon

{

"type": "Polygon",

"coordinates": [

[[7.5,47.5],[7.5,47.6],[7.7,47.6],[7.7,47.5],[7.5,47.5]] // lon, lat

]

}

MultiPolygon

{

"type": "MultiPolygon",

"coordinates": [

[[[8.0,47.4],[8.0,47.6],[8.2,47.6],[8.2,47.4],[8.0,47.4]]], // lon, lat

[[[7.5,47.5],[7.5,47.6],[7.7,47.6],[7.7,47.5],[7.5,47.5]]] // lon, lat

],

"excludeSeaPoints": true,

"fallbackToNearestNeighbour": true

}

The optional Boolean parameter excludeSeaPoints can be set to true, to ignore grid-cells that are located on the sea.

If no grid-cells are within the polygon, the result would be empty.

If fallbackToNearestNeighbour is set to true, the result will select the nearest neighbour grid-cell instead.

### Output Format

The attribute format accepts the following values:

* json: Recommended JSON format (default, if not set)
* csv: CSV format for large amount of locations
* csvTimeOriented: CSV format for long time-ranges
* csvIrregular: CSV format for mixed time-intervals and locations
* geoJson: JSON output to create map with bullet points
* geoJsonHtml: HTML page that embeds a map library and the map json
* netCDF: Recommended binary format for further scientific data analysis

### Time Intervals

Time intervals and timezones can be specified using the ISO8601 format. The timeIntervals attribute is an array of ISO8601 strings. Per default the web-interfaces generate time-intervals with a timezone offset, but without specifying the hour and minute.

{

"timeIntervals": [

"2015-05-01T+00:00/2015-05-02T+00:00",

"2016-05-01T+00:00/2016-05-02T+00:00"

]

}

In the intervals above, 2 full days are selected. For hourly data, the API would return 48 hourly values for each time interval. In the API syntax time-intervals could be specified to select exactly 1 hour:

{

"timeIntervals": [

"2019-01-01T00:00+00:00/2019-01-01T01:00+00:00"

]

}

### Datasets and Variables

The selection of datasets and variables is specified in the attribute queries as an array to select multiple datasets. For each dataset, specified by the domain attribute, multiple weather variable codes can then be chosen.

In this example, three variables are selected from ERA5T and then transformed with two transformations. In the same call, data can be selected from the dataset NEMS12 and transformed individually.

{

"queries": [

{

"domain": "ERA5T",

"gapFillDomain": "NEMSGLOBAL",

"timeResolution": "hourly",

"codes": [

{"code": 11, "level": "2 m above gnd"},

{"code": 52, "level": "2 m above gnd"},

{"code": 157, "level": "180-0 mb above gnd"}

],

"transformations": [

{

"type": "aggregateDaily ",

"aggregation": "mean"

}

]

},

{

"domain": "NEMS12",

"gapFillDomain": null,

"codes": [ ... ],

"transformations": [...]

}

]

}

Attributes for the structure query:

| **Property** | **Type** | **Description** |
| --- | --- | --- |
| domain | String | ERA5T ( kindly use this dataset) |
| gapFillDomain | Optional String | NEMSGLOBAL (kindly use this dataset, dataset to use to fill gaps). |
| timeResolution | String | hourly or daily |
| codes | Array of Codes | Individual selection of weather variables. |
| transformations | Optional array of transformations |  |

Notes:

* timeResolution specified the resolution to read. It can also be set to **daily** although the dataset only offers hourly data to calculate daily aggregations automatically. Aggregations like **monthly** must use transformations.

Once the dataset has been selected, multiple variables at different levels can be encoded into the call. The API is capable of assigning multiple variables per dataset at once. This could improve API call performance because expensive spatial calculations are only performed once.

Attributes for the structure code:

| **Property** | **Type** | **Description** |
| --- | --- | --- |
| code | Integer | Numeric variable code. E.g. **11** for temperature |
| level | String | Level the variable. E.g. **2 m above gnd** |
| aggregation | Optional String | **min, max, mean, sum** to be used with daily aggregations |
|  |  |  |

### List of Variable JSON



### Useful Links

The detailed API Documentation : <https://docs.meteoblue.com/en/weather-apis/dataset-api/dataset-api>

Phython SDK : <https://github.com/meteoblue/python-dataset-sdk>

# Soil API

We have around two datasets which covers the physical and chemical properties of the soil. This data can be consumed using the same **historical API.**

Since the soil data is static, we don’t have any update and the data can be used irrespective of the time duration of the request.

**For the list of variables of the Soil, kindly refer to the section 4.1.6**

### Interpret USDA Texture Classes Values[​](https://docs.meteoblue.com/en/meteo/data-sources/datasets#how-to-interprete-usda-texture-classes-values)

The values returned from SoilGrids -> USDA texture classes are corresponding to the quantity in the following lookup table:

<ColorMap>  
 <ColorMapEntry color="#d5c36b" label="Cl" opacity="1.0" quantity="1" />  
 <ColorMapEntry color="#b96947" label="SiCl" opacity="1.0" quantity="2" />  
 <ColorMapEntry color="#9d3706" label="SaCl" opacity="1.0" quantity="3" />  
 <ColorMapEntry color="#ae868f" label="ClLo" opacity="1.0" quantity="4" />  
 <ColorMapEntry color="#f86714" label="SiClLo" opacity="1.0" quantity="5" />  
 <ColorMapEntry color="#46d143" label="SaClLo" opacity="1.0" quantity="6" />  
 <ColorMapEntry color="#368f20" label="Lo" opacity="1.0" quantity="7" />  
 <ColorMapEntry color="#3e5a14" label="SiLo" opacity="1.0" quantity="8" />  
 <ColorMapEntry color="#ffd557" label="SaLo" opacity="1.0" quantity="9" />  
 <ColorMapEntry color="#fff72e" label="Si" opacity="1.0" quantity="10" />  
 <ColorMapEntry color="#ff5a9d" label="LoSa" opacity="1.0" quantity="11" />  
 <ColorMapEntry color="#ff005b" label="Sa" opacity="1.0" quantity="12" />  
 <ColorMapEntry color="#ffffff" label="NODATA" opacity="0.0" quantity="255" />  
</ColorMap>

where Cl = clay, Si=Silt, Sa=Sand, Lo=Loam.

#### 