

# **Aurinkopaneelien energiantuotantodata Excel-tiedostosta PDF-dokumentiksi ohjelmallisesti**

Lassi Aaltonen, Jami Jokinen, Kasper Kouhia, Tomi Mauno

---

“Aurinkovoima”-ohjelmistoprojektia käsittelevä dokumentti, joka käy läpi projektin suunnittelun ja toteutuksen.

# Aurinkopaneelien energiantuotantodata Excel-tiedostosta PDF-dokumentiksi ohjelmallisesti, 08/11/2020

Lassi Aaltonen, Jami Jokinen, Kasper Kouhia, Tomi Mauno

---

Tämä dokumentti käsittelee Aurinkovoima-projektin suunnittelua ja toteutusta.

Aurinkovoima-projektissa tavoitteena oli luoda ohjelma, joka muuttaa Excel-tiedostosta haetun aurinkopaneelien energiantuotantodatan PDF-dokumentiksi.

PDF-dokumentissa tulee esiintyä energiantuotantodatasta luodut kaaviot. Tulee dokumentissa myös esiintyä energiantuotantodatan mittauspäiviltä säätiedot taulukossa.

Tulee dokumentin myös olla tyylikäs ja helppolukuinen.

## Sisältö

Projektin työtekomenetelmät	3
Ohjelman suunnittelu ja testaus	4
Vaatimusten määrittely ja validaatiotestaus	4
Arkkitehtuurin suunnittelu ja integrointitestaus	4
Moduulien suunnittelu ja moduulitestaus	5
Ohjelman rakenne	5
Python kirjastot	6
Säättöjen haku	6
Ohjelman versiot	7
Ohjelman toiminta ja toiminnot	7
Käyttöliittymä toiminnot	9
PDF-tiedoston luonti toiminnot	10
Muut toiminnot	12
Ohjelman jatkuva tuki ja toiminta	13

## Projektin työtekomenetelmät

Projektissa työskenneltiin eri ketteriä kehitysmenetelmiä hyödyntäen.

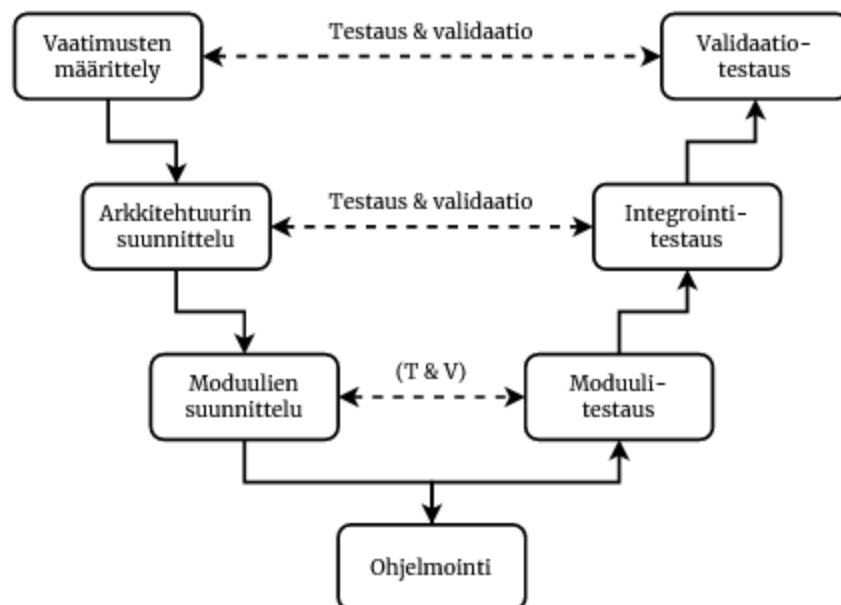
Työtehtävät ja vastuut jaettiin työryhmän jäsenien kesken Scrum-menetelmän mukaan. Työtehtäviä ja niiden tekijöitä pystyttiin muuttamaan tarpeen mukaan.

Ohjelman suunnittelussa, rakentamisessa, ja testauksessa seurattiin V-mallin mukaista prosessikaaviota.

Suunniteltiin siis ohjelman eri osat ja toiminnot määrittelystä aloittaen, siirtyen aina ohjelmassa alemmalle tasolle, ohjelman alimpiin moduuleihin/toimintoihin asti.

Joka tasolle myös suunniteltu eri testausmetodit, joita käytetään testaamaan kyseiselle tasolle suunnittelun mukaisesti toteutetut toiminnot.

Nämä ohjelman suunnittelu-, toteutus-, ja testausvaiheet voidaan esittää nätimmin seuraavanlaisella V:n muotoisella kaaviolla:



## Ohjelman suunnittelu ja testaus

Tämä osio on jaettu V-mallin mukaisesti seuraaviin kolmeen eri alaosiioon:

1. Vaatimusten määrittely ja validaatiotestaus
2. Arkkitehtuurin suunnittelu ja integrointitestaus
3. Moduulien suunnittelu ja moduulitestaus

### Vaatimusten määrittely ja validaatiotestaus

Ohjelman vaatimukset määriteltiin seuraavanlaisiksi:

1. Ohjelman tulee lukea käyttäjän valitsema Excel-tiedosto, ja hakea siitä ohjelmalle olennaisen datan
2. Ohjelman tulee hakea Excel-tiedostosta haettujen päivämäärien säätiedot, jos käyttäjä niin haluaa
3. Ohjelman tulee tulostaa säätiedot ja datan tyylikkääseen, sekä helposti luettavaan PDF-dokumenttiin

Missä ohjelmaan käyttäjän syöttämä Excel-tiedosto sisältää aurinkopaneelien energiantuotantodataa.

Validaatiotestauksessa testattiin, että ohjelma täytti määritellyt vaatimukset.

### Arkkitehtuurin suunnittelu ja integrointitestaus

Tässä osiossa määriteltiin ja suunniteltiin ohjelman arkkitehtuuri ja korkeatasoiset toiminnot.

Voidaan ohjelman korkeatasoiset toiminnot määritellä seuraavanlaisesti:

1. Luo käyttöliittymä, suorita käyttöliittymän eri toimintoja
2. Lue käyttäjän valitseman Excel-tiedoston sisältö, ja rakenna kaavioissa käytettävä data luetusta sisällöstä
3. Hae Excel-tiedostosta haettujen päivämäärien säätiedot, ja rakenna niistä säätaulukossa käytettävä data
4. Luo aurinkopaneelien energiatuotanto kaaviot
5. Luo säätaulukko

6. Luo PDF-tiedostoon asetettavat tekstipätkät
7. Rakenna ja tallenna PDF-tiedosto

Näiden toimintojen toimivuutta käydään läpi tarkemmin seuraavassa osiossa V-mallin prosessia. Voidaan seuraavaa osiota pitää myös edellisten toimintojen alatoimintoina.

Integrintitestauksessa varmistettiin näiden korkeatasoisten toimintojen määrittelyjen mukainen toimivuus.

## **Moduulien suunnittelu ja moduulitestaus**

Moduulien suunnittelussa käytiin läpi arkkitehtuurin suunnittelussa suunniteltuja toimintoja tarkemmin ohjelmassa alemmalla tasolla. Ohjelman tuotantoversio tulee koostumaan näistä moduuleista tai matalatasoisista toiminnoista.

Moduulitestauksessa testattiin suunniteltujen moduulien toimivuus.

## **Ohjelman rakenne**

Aurinkovoima-ohjelman lähdekoodi on Python-ohjelmointikielellä kirjoitettu, jossa käytössä seuraavat kirjastot:

- pandas
- matplotlib
- svglib
- reportlab
- tkinter
- sys
- os
- io
- json
- requests
- datetime
- pyinstaller

## **Python kirjastot**

**Pandas**-kirjastoa käytettiin lukemaan ja muuttamaan Excel-tiedosto Pythonissa luettavaan datamuotoon.

**Matplotlib**-kirjastolla luotiin PDF-tiedostossa esiintyvät kaaviot.

**Svglib**-kirjastoa käyttäen muutettiin kaaviot SVG-kuvamuotoon.

**Reportlab** luo ja rakentaa ohjelman tuotoksena toimivan PDF-tiedoston.

**Tkinter**-kirjastoa käytettiin luomaan ohjelman käyttöliittymä.

**Sys**- ja **os**-kirjastot laajentavat Pythonin käyttöliittymään liittyviä toimintoja. Käytetty tiedostojen sijainteja haettaessa, sekä sulkemaan ohjelma käyttöliittymä suljettua.

**Io** tallentaa kaaviot muistiin, ennen kun ne muutetaan SVG-tiedostoiksi.

**Json**-kirjastoa käytetään lukemaan ja kirjoittamaan ohjelman asetusten tiedostoa.

**Requests**-kirjaston avulla ohjelma pystyy suorittamaan HTTP-pyyntöjä (HTTP GET Requests), joita käyttäen haetaan ohjelmassa käytetyt säätiedot.

**Datetime**-kirjastoa käyttäen voi ohjelma käsitellä päivämääriä helpommin.

**Pyinstaller** pakkaa Python lähdekoodin helpommin suoritettavaan .exe-tiedostoon.

## **Säätietojen haku**

Vaati ohjelma myös ulkoisen palvelun, josta hakea säätietoja PDF-tiedoston säätaulukkoa luotaessa.

Ohjelmassa päädyimme käyttämään ClimaCell-palvelua, tarkemmin ClimaCell Historical Station API:a. Tämä API pystyy hakemaan lähes neljän viikon takaisia säätietoja HTTP-pyyntöissä asetetusta koordinaateista.

ClimaCell Historical Station API hyväksyy maksimissaan 100 kutsua tunnissa.

Ohjelma suorittaa API-kutsun aina seitsemän kertaa viikkoraporttia luodessa, jos käyttäjä on asettanut säätaulukon luonnin päälle. On siis mahdollista suorittaa säätaulukollinen PDF-tiedoston luonti maksimissaan 14 kertaa tunnissa.

Lisää ClimaCellin toiminnasta osoitteessa: <https://www.climacell.co/>

## Ohjelman versiot

Ohjelmasta on myös tehty kaksi eri versiota, joissa hieman erilaiset rakenteet ja toiminnot:

**Perusversio,**  
missä käyttöliittymä ja laaja määrä PDF-dokumentin luontiin liittyviä asetuksia. Ohjelman perusversio pystyy luomaan sekä A4, että A5 kokoisia dokumentteja.

**Express/pikaversio,**  
missä ei ole käyttöliittymää, tai mahdollisuutta käyttäjällä valita Excel-tiedostoa manuaalisesti. Express hakee käyttäjän laitteen Ladatut tiedostot/Downloads-kansiosta uusimman Excel-tiedoston, josta rakentuu A4 kokoinen PDF-dokumentti automaattisesti.

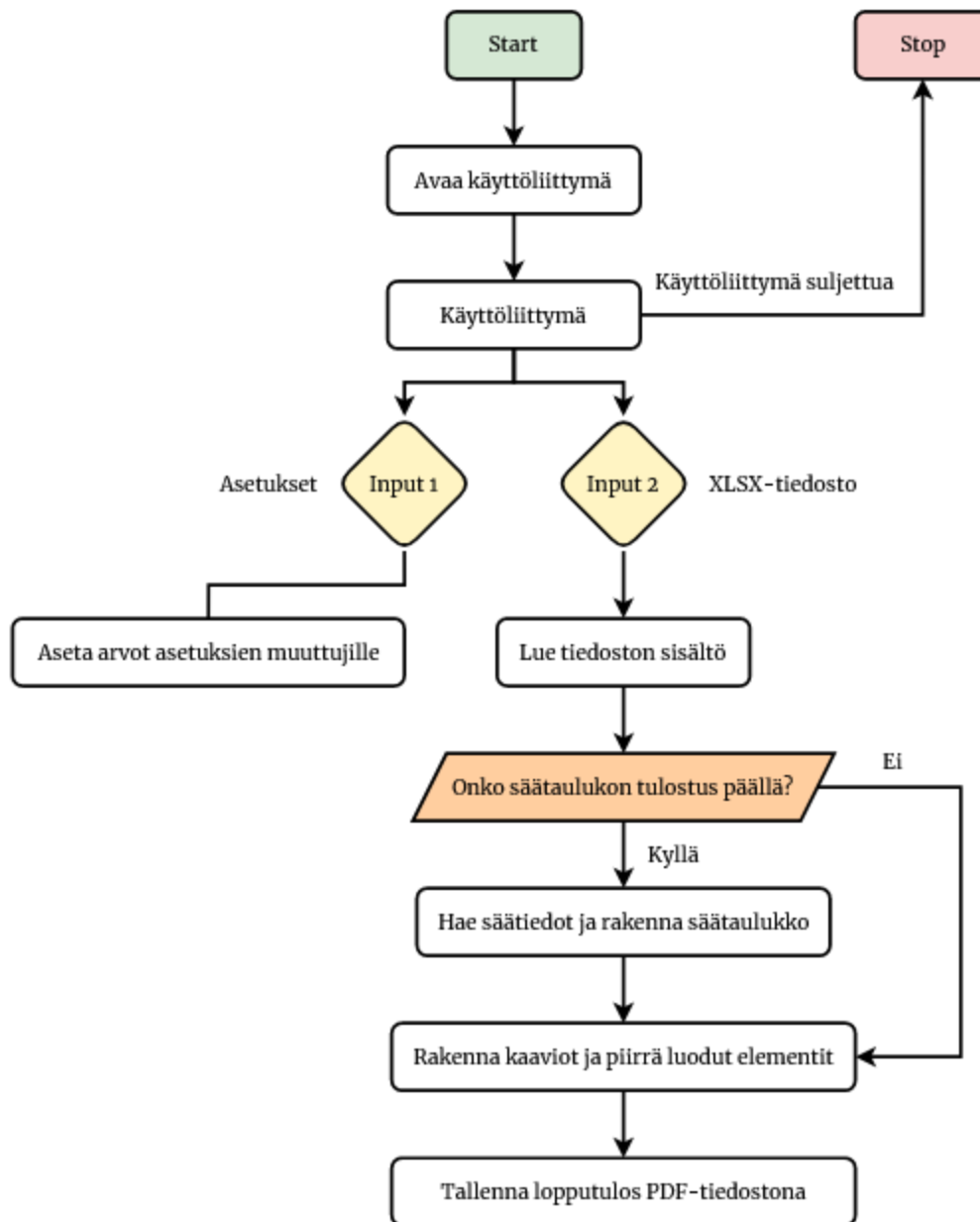
## Ohjelman toiminta ja toiminnot

Tämä osio käsittelee suurimmalta osaltaan perusversion toiminnallisuutta.

Yksinkertaistettuna ohjelman yleinen käyttötapaus suoriutuu ohjelman näkökulmasta seuraavanlaisesti:

1. Avaa käyttöliittymä
2. Odota käyttäjältä syötettä
3. Käyttäjän syötettyä Excel-tiedosto, hae käyttäjän asettamat asetukset, ja luo kyseisten asetusten mukainen PDF-dokumentti
4. Avaa luoto PDF ja tallenna asetukset, jos käyttäjä on asettanut asetusten tallennuksen päälle
5. Kun käyttäjä sulkee käyttöliittymän, lopeta ohjelman suoritus





*Vuokaavio rakenne ohjelman toimintaprosessista.*

Voidaan ohjelman toiminnot jakaa kahteen eri osaan: käyttöliittymä, ja PDF-tiedoston luonti toimintoihin.

Käyttöliittymä toiminnot suoriutuvat heti ohjelman käynnistettyä. Nämä toiminnot rakentavat käyttäjän näkemän käyttöliittymä ikkunan.

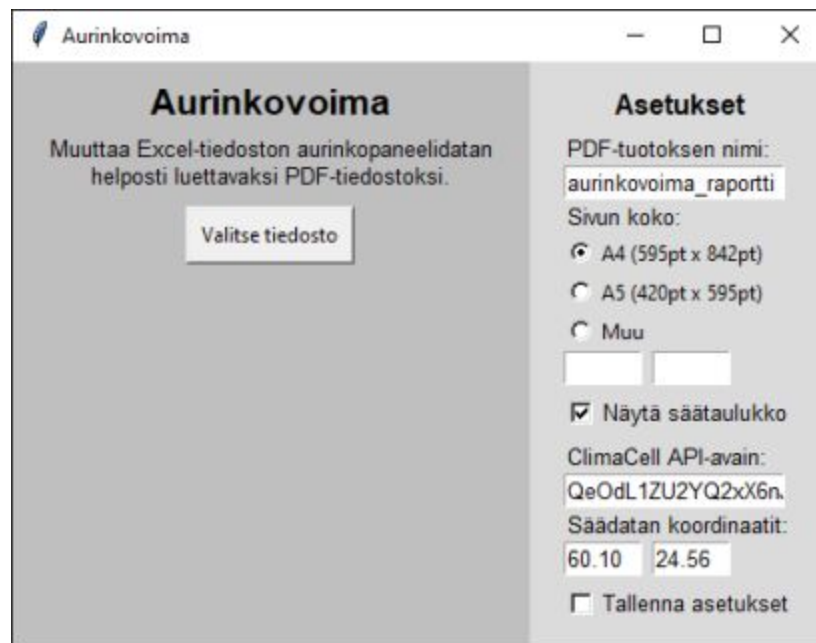
PDF-tiedoston luontiin käytettävät toiminnot suoriutuvat käyttäjän syötettyä Excel-tiedoston.

Näissä toiminnoissa käytetään käyttöliittymässä asetettuja asetuksia luomaan käyttäjän haluaman näköinen PDF-tiedosto.

## Käyttöliittymä toiminnot

Ohjelman käyttöliittymä on rakennettu Pythonin tkinter-kirjastolla.

Käyttöliittymä koostuu tkinter-kirjastolla tehdyistä rungoista, joiden sisälle on pakattu muita tkinter-objekteja (joita kutsutaan myös nimellä "widget").



*Aurinkovoima-ohjelman käyttöliittymä.*

Ohjelman käyttöliittymässä esiintyy hyvin eri tkinter-objektit. Käyttöliittymän ikkunassa suurimpina elementteinä toimivat rungot pääpaneelille ja asetusten sivupaneelille.

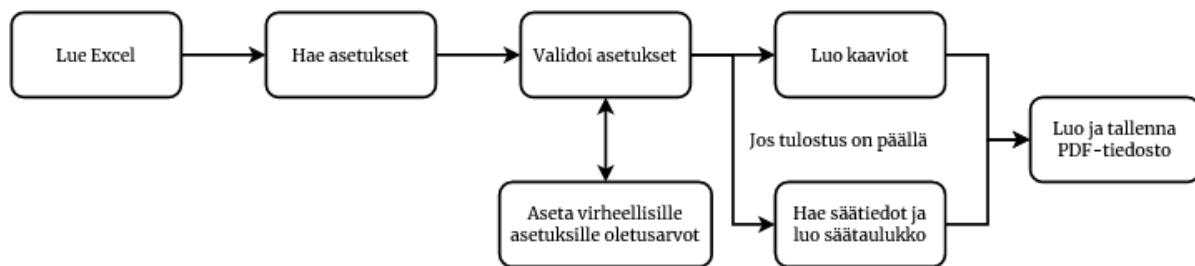
Kyseisten paneelien sisälle on asetettu vielä lisää tkinter-objekteja. Täten lisäten käyttöliittymään tekstiä, painikkeita, ja tekstikenttiä.

Varsinkin asetuksille tarkoitettuun sivupaneeliin on asetettu monenlaisia eri painikkeita ja tekstikenttiä. Näihin elementteihin syötetyt arvot on tallennettu niille asetettuihin muuttujiin. Nämä muuttujat haetaan myöhemmin käyttäjän valittua Excel-tiedoston.

## PDF-tiedoston luonti toiminnot

Käyttäjän valittua Excel-tiedoston ohjelma suorittaa toiminnot käyttäjän asettamien asetusten mukaan.

Voidaan PDF-tiedoston luonti toimintojen kulku ohjelmassa esittää helpoiten seuraavanlaisella kaaviolla:



*PDF-tiedoston luonnin prosessikaavio.*

PDF-tiedostoa luotaessa luetaan ensin käyttäjän syöttämä Excel-tiedoston sisältö. Excel-tiedoston sisällöstä luodaan kolme muuttujaa, jotka vastaavat PDF-tiedostossa tulevien kolmen kaavion käyttämiä dataa/tietoja.

Seuraavaksi ohjelma hakee käyttöliittymässä käyttäjän asettamien asetusten arvot. Ohjelma validoi asetusten arvot, korvaten virheelliset arvot asetusten oletusarvoilla.

Asetuksista vaikutusvaltaisimpana voidaan pitää PDF-tiedoston sivun kokoa, joka vaikuttaa sekä PDF-tiedoston sivun leveyteen ja pituuteen, että itse sivun sisällä oleviin elementteihin. Tehden näin ohjelman luomasta PDF-tiedoston sisällöstä responsiivisen jokaisella käyttäjän asettamalla sivun koolla.

Tämä responsiivisuus on suoritettu luomalla yksikkö, jonka kokoa käytetään pohjana kaikkia PDF-tiedoston elementtejä luodessa.

Ohjelmassa käytetty yksikkö on laskettu seuraavanlaisesti:

$$\frac{X + Y}{2} \cdot 0.015$$

*Missä X on sivun leveys ja Y on sivun pituus.*

A4-kokoisella sivulla olisi yksikkö siis laskettu näin:

$$\frac{595 + 842}{2} \cdot 0.015 = 10.7775$$

*Jolloin yksikkö on kooltaan 10.7775pt.*

Yksikön koko asetettua luo ohjelma PDF-tiedostossa esiintyvät tekstipätkät ja kaaviot yksikön koon mukaan.

Eri elementtejä asettaessa PDF:ään käytetään reportlab-kirjaston koordinaattisysteemiä, missä jokainen sivun piste (pt) asettuu x- ja y-koordinaatistoon. Reportlab asettaa koordinaateissa sijainnin (0, 0) sivun vasempaan alakulmaan. Jolloin A4 kokoisen sivun oikean yläkulman koordinaatit ovat (595, 842).

Muutettiin kuitenkin Y-akseli toimimaan ohjelmassa toisin päin, jotta sen käyttö olisi intuitiivisempaa. PDF-tiedoston rakentaminen sujui myös helpommin ylhäältä alaspäin.

Asetetaan säätaulukkokin paikalleen PDF:ssä tällä systeemillä. Säätaulukon sisältö on haettu ClimaCell API-kutsulla. API-kutsussa käytetään käyttäjän asettamia koordinaatteja ja API-avainta. Haettu säädata vastaa syötetyn Excel-tiedoston sisältämiä päivämääriä, paitsi jos Excel-tiedosto sisältää enemmän kuin seitsemän päivämäärää.

Säätaulukossa esiintyvät kuvakkeet riippuvat haetusta säädatasta.

PDF-tuotos tallentuu reportlab-kirjastoa käyttäen.

## Muut toiminnot

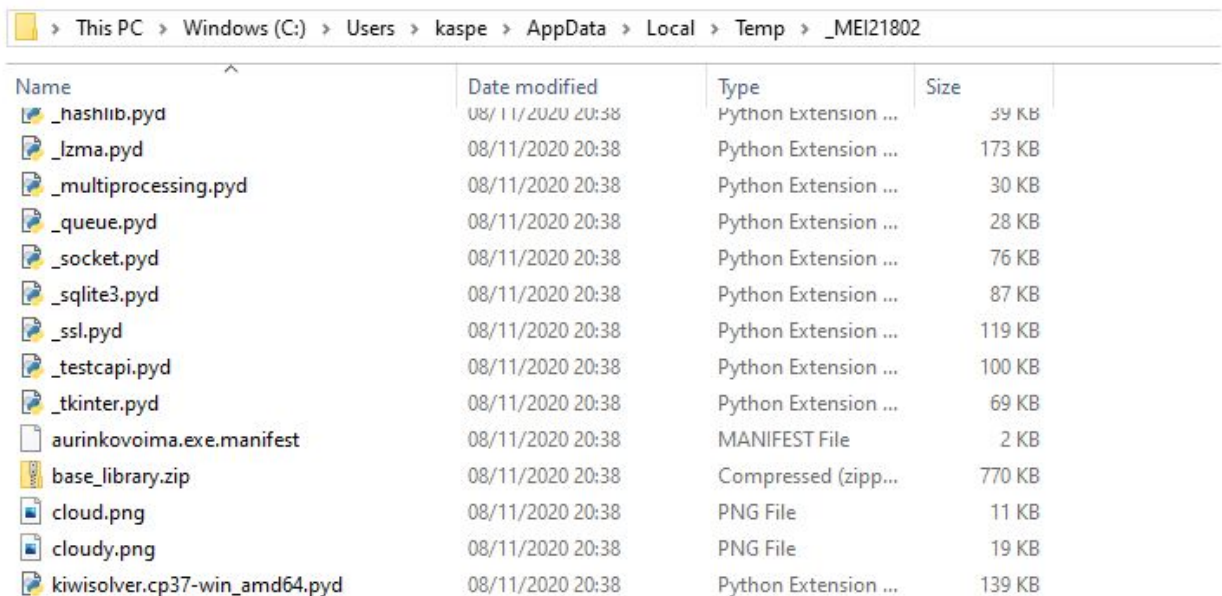
Nämä toiminnot eivät sopineet kunnolla kumpaankaan aikaisemmista toimintokategorioista.

Käyttäjä ei tule koskaan huomaamaan näitä toimintoja, mutta ohjelma vaatii niitä toimimaan kunnolla.

Näistä sekalaisista toiminnoista tärkein on toiminto, joka antaa ohjelman hakea säätaulukossa käytettävät kuvakkeet sekä pakkaamattomana, että pakattuna ohjelmana.

Kun ohjelma suoritetaan sen pakkaamattomana Python-tiedosto muodossa, haetaan kuvakkeet images-kansiosta.

Kun ohjelma suoritetaan pakattuna .exe-tiedostona, haetaan kuvakkeet ohjelman luomasta väliaikaisesta kansiota.



Name	Date modified	Type	Size
_hashlib.pyd	08/11/2020 20:38	Python Extension ...	39 KB
_lzma.pyd	08/11/2020 20:38	Python Extension ...	173 KB
_multiprocessing.pyd	08/11/2020 20:38	Python Extension ...	30 KB
_queue.pyd	08/11/2020 20:38	Python Extension ...	28 KB
_socket.pyd	08/11/2020 20:38	Python Extension ...	76 KB
_sqlite3.pyd	08/11/2020 20:38	Python Extension ...	87 KB
_ssl.pyd	08/11/2020 20:38	Python Extension ...	119 KB
_testcapi.pyd	08/11/2020 20:38	Python Extension ...	100 KB
_tkinter.pyd	08/11/2020 20:38	Python Extension ...	69 KB
aurinkovoima.exe.manifest	08/11/2020 20:38	MANIFEST File	2 KB
base_library.zip	08/11/2020 20:38	Compressed (zipp...	770 KB
cloud.png	08/11/2020 20:38	PNG File	11 KB
cloudy.png	08/11/2020 20:38	PNG File	19 KB
kiwisolver.cp37-win_amd64.pyd	08/11/2020 20:38	Python Extension ...	139 KB

*Ohjelman luoma väliaikainen kansio, sekä osa sen sisällöstä.*

Toinen näistä sekalaisista toiminnoista käsittelee päivämääriä ja niiden eri muotoja.

Ohjelma saa Excel-tiedostosta päivämäärät datetime-muodossa, joka voidaan helposti muokata eri tekstipätkiksi, joita käytetään eri osissa ohjelmaa.

Säätietoja haettaessa käytetään esimerkiksi seuraavanlaista päivämäärän muotoa:

$(YY) - (MM) - (DD)T(HH) - (MI) - (SS)Z$

Missä YY = vuosi, MM = kuukausi, DD = päivä, HH = tunti, MI = minuutti, SS = sekunti  
Joissa jokaisessa edeltävät nollat.

Muutetaan myös päivämäärät eri muotoihin esitettäväksi PDF-tiedostossa.

Voi käyttäjä myös tallentaa PDF-tiedostoa tehdessä asetetut asetukset. Tämä suoriutuu helposti kirjoittamalla json-kirjastolla json-muotoiseen tiedostoon asetuksista haetut arvot.

## Ohjelman jatkuva tuki ja toiminta

Ohjelman jatkuva toiminta vaatii ClimaCell API-avaimen vaihtoa joka vuosi.

Vanhentunut API-avain ei tule estämään ohjelman muita toimintoja, kunhan käyttäjä on asettanut säätaulukon tulostuksen pois päältä.

ClimaCell API ei myöskään saata pystyä hakemaan säätietoja neljää viikkoa taaempaa. Ihme kyllä, testauksen aikana ClimaCell API palautti sää tiedot monen kuukauden takaisilta päiviltä.