



# Future solar power generation application system

Source: <https://www.ferraxegalia.es/Fri-30-Dec-2016-2218.html>

Website: <https://www.ferraxegalia.es>

This PDF is generated from: <https://www.ferraxegalia.es/Fri-30-Dec-2016-2218.html>

Title: Future solar power generation application system

Generated on: 2026-02-04 07:24:17

Copyright (C) 2026 GALICIA CONTAINERS. All rights reserved.

For the latest updates and more information, visit our website: <https://www.ferraxegalia.es>

-----

The class template `std::future` provides a mechanism to access the result of asynchronous operations: An asynchronous operation (created via `std::async`, ...

The error: `SyntaxError: future feature annotations is not defined` usually related to an old version of python, but my remote server has Python3.9 and to verify it - I also added it ...

Checks if the future refers to a shared state. This is the case only for futures that were not default-constructed or moved from (i.e. returned by `std::promise::get_future()`, ...

Transfers the shared state of `*this`, if any, to a `std::shared_future` object. Multiple `std::shared_future` objects may reference the same shared state, which is not possible with ...

If the future is the result of a call to `std::async` that used lazy evaluation, this function returns immediately without waiting. This function may block for longer than ...

```
future (const future & ) = delete; ~future (); future & operator =(const future & ) = delete; future & operator =(future & & ) noexcept; shared_future &lt;R> share () noexcept; // ...
```

Unlike `std::future`, which is only moveable (so only one instance can refer to any particular asynchronous result), `std::shared_future` is copyable and multiple shared future ...

If the future is the result of a call to `async` that used lazy evaluation, this function returns immediately without waiting. The behavior is undefined if `valid()` is false before the call ...

The `get` member function waits (by calling `wait()`) until the shared state is ready, then retrieves the value

stored in the shared state (if any). Right after calling this function, valid ...

Specifies state of a future as returned by `wait_for` and `wait_until` functions of `std::future` and `std::shared_future`. Constants

Web: <https://www.ferraxegalia.es>

