

# Future energy storage projects



## Future energy storage projects



[The Future of Energy Storage , MIT Energy Initiative](#)

As the electric grid grows more complex, battery energy storage systems are proliferating. Here's how developers can succeed in a rapidly

### Long Duration Energy Storage Program

As the deployment of intermittent renewable energy sources accelerates and the frequency of extreme weather events increases due to climate change, there is a growing need for



### std::future::~~future

Releases any shared state. This means: If the current object holds the last reference to its shared state, the shared state is destroyed. The current object gives up its reference to its shared

### std::future

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



### Standard library header (C++11)

```
future (const future &) = delete; ~future ();
future & operator =(const future &) = delete;
future & operator =(future &&) noexcept;
shared_future share () noexcept; // retrieving the
```

value

### **std::future::wait\_until**

wait\_until waits for a result to become available. It blocks until specified timeout\_time has been reached or the result becomes available, whichever comes first. The return value indicates why



### **std::future::wait\_for**

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 timeout\_duration due to

### **std::future::get**

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 () is false.



## **ENERGY STORAGE PROJECTS**

Accelerated by DOE initiatives, multiple tax credits under the Bipartisan Infrastructure Law and Inflation Reduction Act, and decarbonization goals

### **std::future\_status**

Specifies state of a future as returned by wait\_for and wait\_until functions of std::future and std::shared\_future. Constants





### **std::shared\_future**

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 objects

### **std::future::valid**

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()`),



## Contact Us

---

For catalog requests, pricing, or partnerships, please visit:  
<https://european-startups.eu>