

An abstract composition on a dark gray background. It features several diagonal lines: a white line in the upper left, a light gray line in the lower left, and a bright pink line in the lower right. The word 'Substrate' is written in a bold, orange, sans-serif font, positioned between the white and light gray lines. The word 'River' is written in the same font and color, positioned between the light gray and pink lines.

**Substrate**

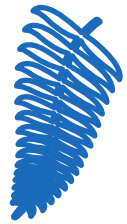
**River**



**Basic-004**

# Storing a Storage Mapping

<https://substrate.dev/substrate-collectables-workshop/#/1/storage-mapping>



Substrate的特殊类型

AccountId

Balance

Hash



使用

T::Type



Trait修改: balances继承了原来的Traits

```
pub trait Trait: balances::Trait {}
```



## 声明Mapping

Map (key, value)

SomeValue **get**(some\_value\_getter): map u32 => u32;

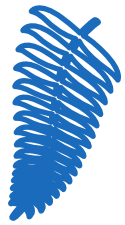
MyValue: map T::AccountId => u32;

OwnedKittyCount get(owned\_kitty\_count): map T::AccountId=> **u32**;



## 导入StorageMap类型

support::StorageMap



## 使用Mapping

```
/// Get the prefix key in storage.
fn prefix() -> &'static [u8];

/// Get the storage key used to fetch a value corresponding to a specific key.
fn key_for(x: &K) -> Vec<u8>;

/// true if the value is defined in storage.
fn exists<S: Storage>(key: &K, storage: &S) -> bool {
    storage.exists(&Self::key_for(key)[..])
}

/// Load the value associated with the given key from the map.
fn get<S: Storage>(key: &K, storage: &S) -> Self::Query;

/// Take the value under a key.
fn take<S: Storage>(key: &K, storage: &S) -> Self::Query;

/// Store a value to be associated with the given key from the map.
fn insert<S: Storage>(key: &K, val: &V, storage: &S) {
    storage.put(&Self::key_for(key)[..], val);
}

/// Remove the value under a key.
fn remove<S: Storage>(key: &K, storage: &S) {
    storage.kill(&Self::key_for(key)[..]);
}

/// Mutate the value under a key.
fn mutate<R, F: FnOnce(&mut Self::Query) -> R, S: Storage>(key: &K, f: F, storage: &S) -> R;
```



例子

增加

```
<SomeValue<T>>::insert(key, value);
```

查询

```
let my_value = <SomeValue<T>>::get(key);  
let also_my_value = Self::some_value_getter(key);
```



## 备注

视频中对`T::Type`中的解释不准确；大家可以去看下Rust对T的讲解。

The image features a dark gray background with several diagonal lines. A white line starts from the left edge and extends towards the top right. A second white line is parallel to the first, starting further to the right and also extending towards the top right. A gray line starts from the bottom left and extends towards the top right. A red line starts from the bottom left and extends towards the top right, positioned below the gray line. The text 'Substrate' is written in a bold, orange font, centered horizontally between the two white lines.

**Substrate**

**River**

**Thanks**