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;
```

https://substrate.dev/rustdocs/v1.0/srml_support/storage/trait.StorageMap.html

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。

Substrate

River

Thanks