SingleSelect 单选字段

类型定义 `ISingleSelectField` ，使用方法示例：

```typescript
const singleSelectField = await table.getField<ISingleSelectField>(fieldId);
```

其中字段值的类型定义为：

```typescript
// 单选值类型定义
type IOpenSingleSelect = {
  id: string; // 选项 id
  text: string;
};

type SingleSelectTransformVal = string | IOpenSingleSelect;
```

createCell

创建一个单选字段的 `Cell` 。

```typescript
createCell: (val: SingleSelectTransformVal) => Promise<ICell>;
```

示例

```typescript
await singleSelectField.createCell('test option');
```

getCell

获取指定记录对应的 `Cell` 单元格。

```typescript
getCell: (recordOrId: IRecordType | string) => Promise<ICell>;
```

示例

```typescript
const table = await bitable.base.getActiveTable();
const recordList = await table.getRecordList();

const cell = await singleSelectField.getCell(recordList[0]);
```

setValue

设置指定单元格的值。

```typescript
setValue: (recordOrId: IRecordType | string, val: SingleSelectTransformVal) => Promise
```

示例

```typescript
const table = await bitable.base.getActiveTable();
const recordIdList = await table.getRecordIdList();

await singleSelectField.setValue(recordIdList[0], 'option_id'); // 传入选项 id
```

getValue

获取指定单元格的值。

```typescript
getValue: (recordOrId: IRecordType | string) => Promise<IOpenSingleSelect>;
```

示例

```typescript
const table = await bitable.base.getActiveTable();
const recordIdList = await table.getRecordIdList();

const cellValue = await singleSelectField.getValue(recordIdList[0]);
```

addOption

新增选项，可指定选项名称和颜色。

```typescript
addOption: (name: string, color?: number) => Promise<IFieldRes>;
```

示例

```typescript
await singleSelectField.addOption('new option');
```

addOptions

新增多个选项，可指定选项名称和颜色。

```typescript
addOptions: (optionList: { name: string, color?: number }[]) => Promise<IFieldRes>;
```

示例

```typescript
await singleSelectField.addOptions([
  {
    name: 'new option 1',
  },
  {
    name: 'new option 2',
  }
]);
```

getOptions

获取所有的选项，其中 `ISelectFieldOption` 的类型定义为：

```typescript
getOptions: () => Promise<ISelectFieldOption[]>;

interface ISelectFieldOption {
  id: string;
  name: string;
  color: number;
}
```

示例

```typescript
await singleSelectField.getOptions();
```

deleteOption

通过选项 `id` 或者 `name` 删除选项。

```typescript
deleteOption: (idOrName: string) => Promise<IFieldRes>;
```

示例

```typescript
const options = await singleSelectField.getOptions();

await singleSelectField.deleteOption(options[0].id);
```

setOption

通过选项 `id` 或者 `name` 设置选项，其中 `OptionConfig` 的类型定义为：

```typescript
setOption: (nameOrId: string, config: OptionConfig) => Promise<IFieldRes>;

export type OptionConfig = {
  name?: string;
  color?: number;
};
```

示例

```typescript
const options = await singleSelectField.getOptions();

await singleSelectField.setOption(options[0].id, {
  name: 'modify option'
});
```

getOptionsType

获取选项类型，其中 `SelectOptionsType` 的类型定义为:

```typescript
getOptionsType: () => Promise<SelectOptionsType>;

enum SelectOptionsType {
  STATIC, // 自定义选项
  DYNAMIC, // 引用选项
}
```

示例

```typescript
await singleSelectField.getOptionsType();
```

setOptionsType

设置选项类型，其中 `SelectOptionsType` 的类型定义为:

```typescript
setOptionsType: (type: SelectOptionsType) => Promise<IFieldRes>;

enum SelectOptionsType {
  STATIC, // 自定义选项
  DYNAMIC, // 引用选项
}
```

示例

```typescript
await singleSelectField.setOptionsType(SelectOptionsType.STATIC);
```

Last updated: 2023/12/5 20:47

Previous page
文本字段

Next page
多选字段