

# 登壇者紹介

株式会社メルカリやいくつかのスタートアップでソフトウェア開発に従事し、その後2018年にソフトウェアの設計、開発、およびそれらの支援を行うインスタンスゼロ株式会社を共同創業。2020年にLINE株式会社に入社。LIFF SDKを担当しています。



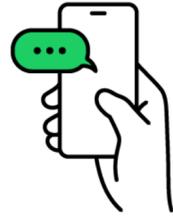
**佐藤 信吾**

LINE株式会社 / UIT Dev2 エンジニアリングマネージャ / ソフトウェアエンジニア

# 認知から完成まで



認知: LIFFとは?



手元でLIFFの  
機能を試してみる



機能を実装する



将来のために  
拡張性をもたせる

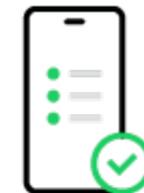
ドキュメントを  
読む



LIFFの  
開発環境をつくる



デバッグツールを  
つかう

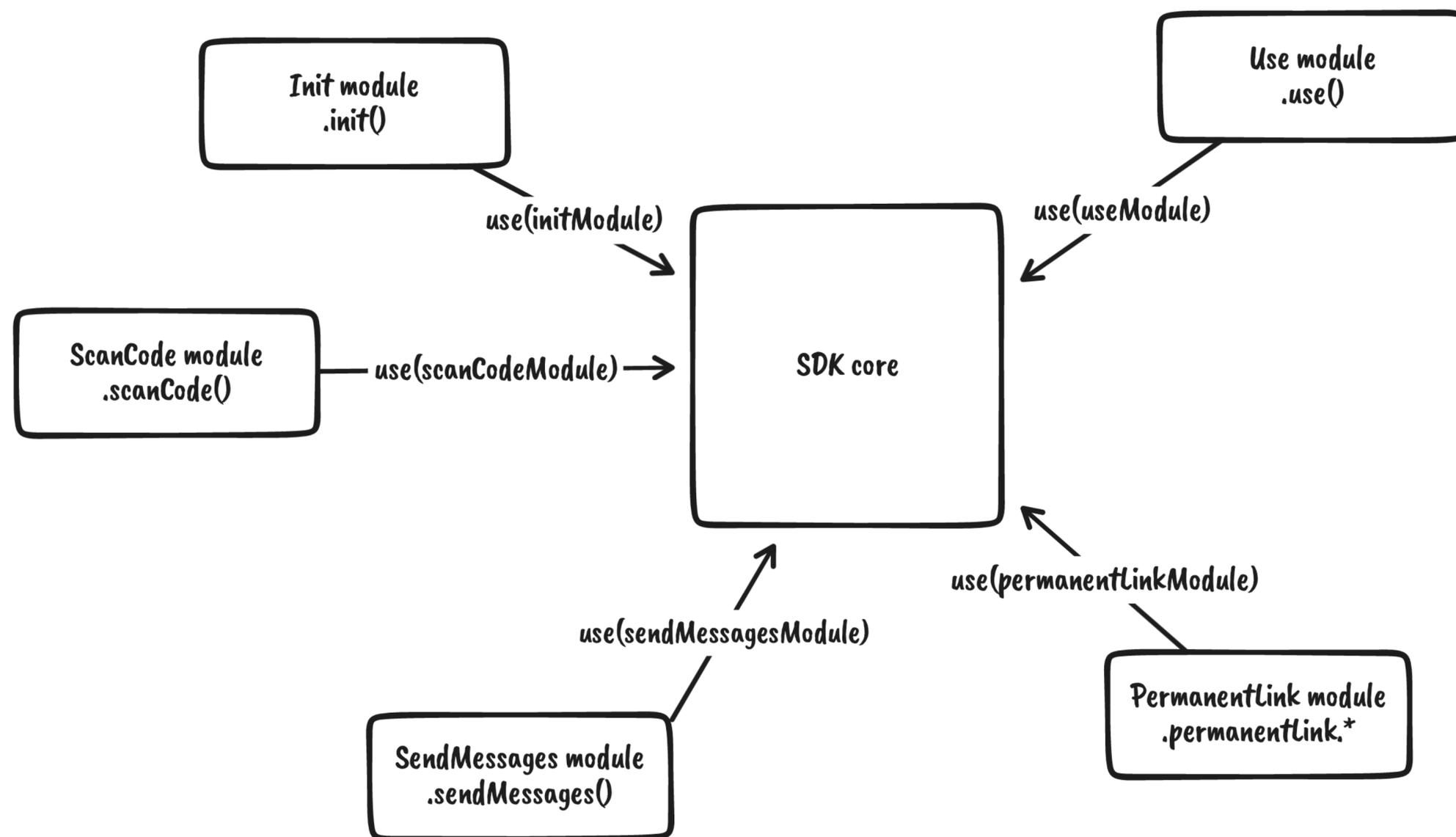


# LIFFプラグイン



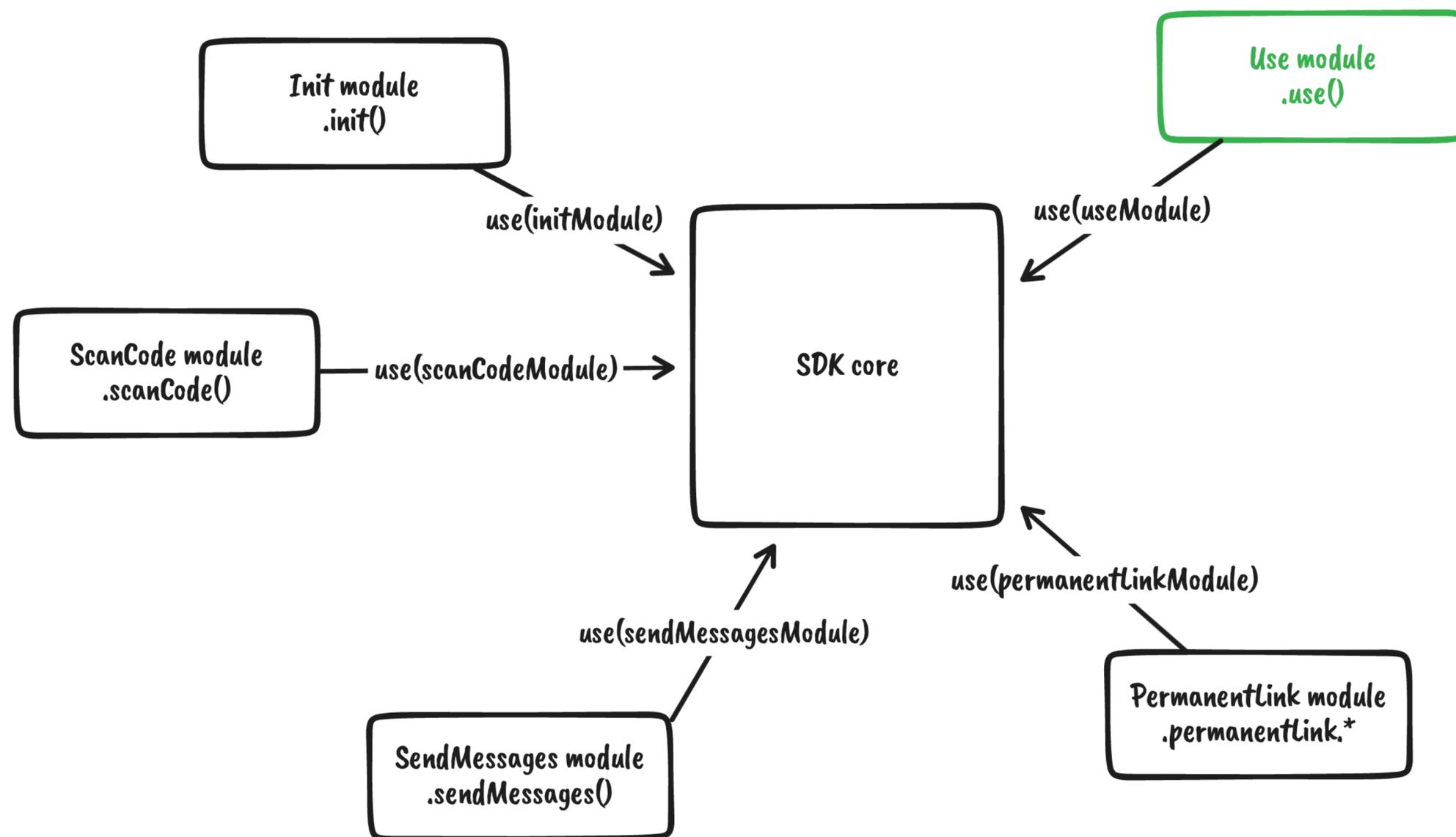
# LIFFプラグイン

## アーキテクチャ



# LIFFプラグイン

## アーキテクチャ



# LIFFプラグイン

## 機能

要素①

SDKオブジェクトに独自ネームスペースの追加

要素②

SDKのAPIへのコールバック関数登録

要素③

プラグイン独自のコールバックイベントの作成

# LIFFプラグイン

```
liff.use(liffPlugin, options);
```

# LIFFプラグイン

## SDKオブジェクトに独自ネームスペースの追加

```
liff.use({
  name: 'greet',
  install() {
    return {
      hello(name = 'World') {
        console.log(`Hello, ${name}!`);
      },
    };
  },
});

liff.$greet.hello(); // Hello, World!
```

# LIFFプラグイン

## SDKのAPIへのコールバック関数登録

```
liff
  .use({
    name: 'simpleHooks',
    install({ hooks }) {
      hooks.init.before(async () => console.log('before hook is called'));
      hooks.init.after(async () => console.log('after hook is called'));
    },
  })
  .init({ liffId: 'your_liff_id' });
// before hook is called
// after hook is called
```

# LIFFプラグイン

## プラグイン独自のコールバックイベントの作成

```
import { SyncHook, AsyncHook } from '@lifff/hooks';

class GreetPlugin {
  name = 'greet';
  hooks = {
    helloBefore: new SyncHook(),
    helloAfter: new AsyncHook(),
  };
  install() {
    return {
      hello: (name = 'World') => {
        this.hooks.helloBefore.call();
        console.log(`Hello, ${name}!`);
        this.hooks.helloAfter.call();
      },
    };
  }
}
```

# LIFFプラグイン

## プラグイン独自のコールバックイベントの作成

```
liff
  .use(new GreetPlugin())
  .use({
    name: 'simpleHook',
    install({ hooks }) {
      hooks.$greet.helloBefore(
        () => console.log('helloBefore is called')
      );
      hooks.$greet.helloAfter(
        async () => console.log('helloAfter is called')
      );
    },
  })
  .$greet.hello('LIFF Plugin');
// helloBefore is called
// Hello, LIFF Plugin!
// helloAfter is called
```

# LIFFプラグイン

```
const greetPlugin = {
  name: 'greet',
  install(context, greet = 'Hello') {
    return {
      hello(name = 'World') {
        console.log(`${greet}, ${name}!`);
      },
    };
  },
};
```

```
liff.use(greetPlugin, 'Hi');
liff.$greet.hello(); // Hi, World!
```

# LIFFプラグイン

```
install({ lifff }) {  
    lifff.getVersion(); // => 2.19.1  
}
```

# LIFFプラグイン

```
class DeepDivePlugin {
  name: string;
  constructor() {
    this.name = 'deepdive';
  }
  install(context: LiffPluginContext) {
    context.hooks.init.after(this.initAfter);
    return {
      shareMessages: this.shareMessages,
      readQrCode: this.readQrCode
    };
  }
  initAfter() {
    if (!liff.isLoggedIn()) {
      liff.login();
    }
    return Promise.resolve();
  }
  shareMessages(): Promise<ShareTargetPickerResult | void> {
    // ...
  }
  readQrCode(): Promise<void> {
    // ...
  }
}
```

# LIFFプラグイン

```
liff.use(new DeepDivePlugin());  
  
// liff.login() called after liff.init() automatically  
// and you can use the following:  
// liff.$deepdive.shareMessages()  
// liff.$deepdive.readQrCode()
```