

Ubuntu 22.04
CPU x86_64
Electron git Electron

aki

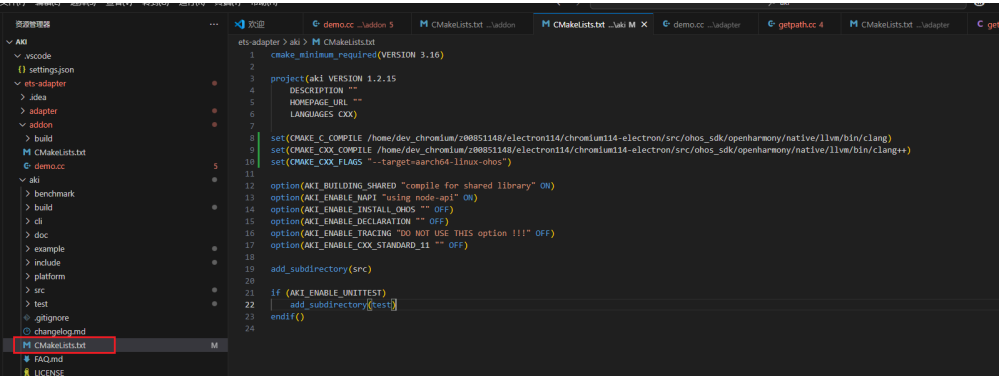
1 git aki

git clone https://gitee.com/openharmony-sig/aki.git

2 cmake

aki cmake clang clang++

set(CMAKE_C_COMPILE /home/dev_chromium/z00851148/electron114/chromium114-electron/src/ohos_sdk/openharmony/native/llvm/bin/clang)
set(CMAKE_CXX_COMPILE /home/dev_chromium/z00851148/electron114/chromium114-electron/src/ohos_sdk/openharmony/native/llvm/bin/clang)
set(CMAKE_CXX_FLAGS "--target=aarch64-linux-ohos")



3 clang clang++

export CC="/home/dev_chromium/z00851148/electron114/chromium114-electron/src/ohos_sdk/openharmony/native/llvm/bin/clang --target=aarch64-linux-ohos"
export CXX="/home/dev_chromium/z00851148/electron114/chromium114-electron/src/ohos_sdk/openharmony/native/llvm/bin/clang++ --target=aarch64-linux-ohos"

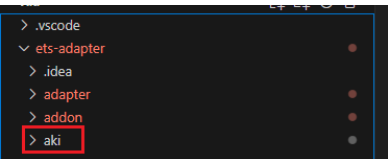
4

build/src libaki_jsbind.so

cd aki
mkdir build
cd build
cmake ../ -DCMAKE_BUILD_TYPE=Release
make

adapter

1



aki git aki

2. adapter demo

```

1  ets-adapter > adapter > demo.cc > getNativeContext(napi_env, napi_callback_info)
2  #include "third_party/electron_node/src/node.h"
3  #include "openharmony/native/sysroot/usr/include/napi/native_api.h"
4  #include "openharmony/native/sysroot/usr/include/js_native_api_types.h"
5  #include "aki/jsbind.h"
6  #include <cstring.h>
7  #include <stdio.h>
8  #define DECLARE_NAPI_METHOD(name, func) { name, 0, func, 0, 0, napi_default, 0 }
9
10 using v8::FunctionCallbackInfo;
11 using v8::Isolate;
12 using v8::Local;
13 using v8::NewStringType;
14 using v8::Object;
15 using v8::String;
16 using v8::Value;
17
18 static napi_value getNativeContext(napi_env env, napi_callback_info info) {}
19
20 napi_value exports;
21 if (napi_create_object(env, &exports) != napi_ok) {
22     napi_throw_error(env, nullptr, "napi_create_object failed");
23     return exports;
24 }
25
26 aki::JSBind::BindSymbols(env, exports);
27 return exports;
28
29 static napi_value Initialize(napi_env env, napi_value exports) {
30     if (env == nullptr || exports == nullptr) {
31         return exports;
32     }
33     napi_property_descriptor desc[] = {
34         DECLARE_NAPI_METHOD("getNativeContext", getNativeContext)
35     };
36     if (napi_define_properties(env, exports, sizeof(desc) / sizeof(desc[0]),
37         desc) != napi_ok) {
38         return exports;
39     }
40     return exports;
41 }
42
43 static napi_module adaptertestModule = {
44     .nm_version = 1,
45     .nm_flags = 0,
46     .nm_filename = nullptr,
47     .nm_register_func = Initialize,
48     .nm_module_name = "adaptertest",
49     .nm_priv = ((void*)nullptr),
50     .reserved = {nullptr},
51 };
52
53 extern "C" __attribute__((constructor)) void RegisterAdapterTestModule(void) {
54     napi_module_register(&adaptertestModule);
55 }

```

3. getPatch.cc

```

1  ets-adapter > adapter > getPatch.cc > PathAdapter > GetDir(std::string R)
2  #include <functional>
3  #include "aki/jsbind.h"
4  namespace PathAdapter {
5  std::string GetDir(std::string& getDirFuncName) {}
6
7  // 同步方式
8  if (auto getDirFunc = aki::JSBind::GetJSFunction(getDirFuncName)) {
9      auto path = getDirFunc->Invoke<std::string>();
10     return path;
11 }
12
13 return "";
14
15 // 异步方式
16 // std::promise<std::string> promise;
17 // std::function<void(std::string)> callback = [&promise](std::string path) {
18 //     promise.set_value(path);
19 // };
20
21 // If (auto jsFunc = aki::JSBind::GetJSFunction(getDirFuncName)) {
22 //     jsFunc->Invoke<void>(callback);
23 // } else {
24 //     return "";
25 // }
26
27 // auto future = promise.get_future();
28 // auto status = future.wait_for(std::chrono::seconds(1));
29 // If (status == std::future_status::timeout) {
30 //     return "timeout";
31 // }
32 // return future.get();
33
34 }
35

```

4. CMakeLists

```

# Project Name
SET(TARGETFILENAME "adaptertest")
PROJECT(${TARGETFILENAME})

# CMake minimum version requirement setting
cmake_minimum_required(VERSION 3.8)

# set electron path
set(ELE_PATH /home/dev_chromium/z00851148/electron114/chromium114-electron)

# ohos
set(CMAKE_C_COMPILER ${ELE_PATH}/src/ohos_sdk/openharmony/native/llvm/bin/clang)
set(CMAKE_CXX_COMPILER ${ELE_PATH}/src/ohos_sdk/openharmony/native/llvm/bin/clang++)
set(CMAKE_CXX_FLAGS "--target=aarch64-linux-ohos")

#
# ohos
include_directories(${ELE_PATH}/src/)
include_directories(${ELE_PATH}/src/third_party/electron_node/deps/v8/include/)
include_directories(${ELE_PATH}/src/ohos_sdk/)
include_directories(${CMAKE_CURRENT_SOURCE_DIR}/../aki/include)

message(${CMAKE_CURRENT_SOURCE_DIR})

set(src demo.cc getpath.cc)
set(headers getpath.h)

#
#add_executable(${TARGETFILENAME} demo.cpp)

#
#ADD_LIBRARY(${TARGETFILENAME} STATIC demo.cpp)

#

```

```

ADD_LIBRARY (${TARGETFILENAME} SHARED
    ${src}
)
target_compile_features(${TARGETFILENAME} PUBLIC cxx_std_17)
target_compile_definitions(${TARGETFILENAME} PUBLIC JSBIND_USING_NAPI=1)
target_compile_definitions(${TARGETFILENAME} PUBLIC AKI_BUILDING_SHARED=1)

#
# -ohos
target_link_libraries(${TARGETFILENAME} PUBLIC ${CMAKE_CURRENT_SOURCE_DIR}/../aki/build/src/libaki_jsbind.so)
target_link_libraries(${TARGETFILENAME} PUBLIC ${ELE_PATH}/src/out/musl_64/libelectron.so)

#
# .node
# SET_TARGET_PROPERTIES(${TARGETFILENAME} PROPERTIES SUFFIX ".node")

```

5.

libadaptestest.so

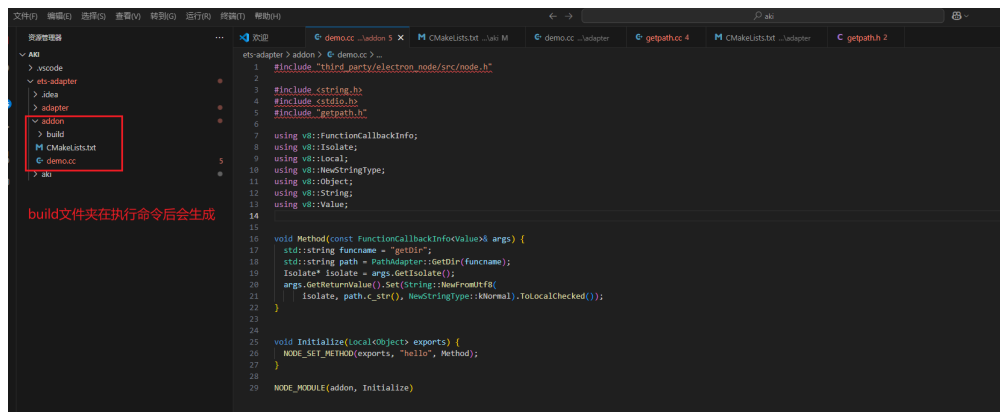
```

mkdir build
cd build
cmake ../
make

```

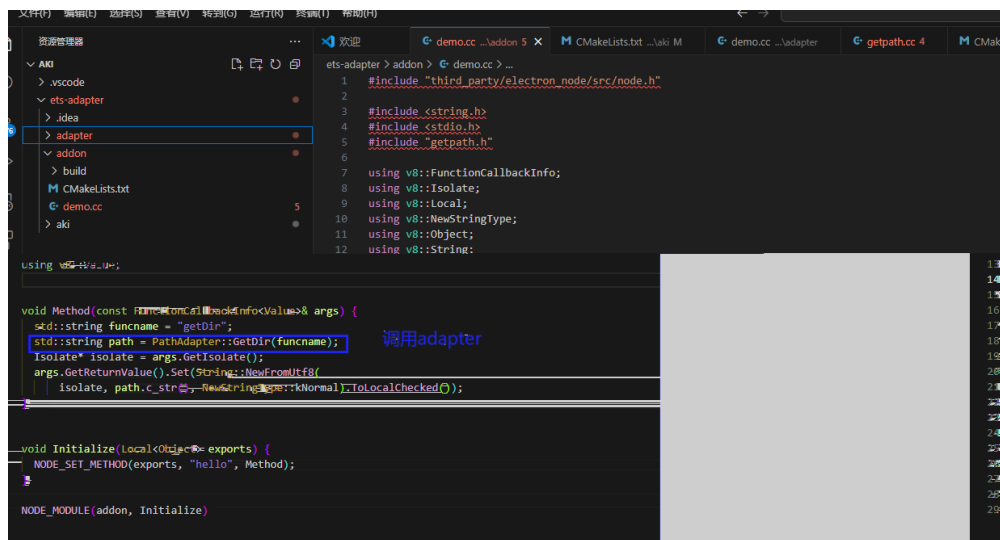
addon

1



2

adapter



3 CMakeLists

```

# Project Name
SET(TARGETFILENAME "addon")
PROJECT(${TARGETFILENAME})

# CMake minimum version requirement setting
cmake_minimum_required(VERSION 3.8)

# set electron path
set(ELE_PATH /home/dev_chromium/z00851148/electron114/chromium114-electron)

# ohos
set(CMAKE_C_COMPILER ${ELE_PATH}/src/ohos_sdk/openharmony/native/llvm/bin/clang)
set(CMAKE_CXX_COMPILER ${ELE_PATH}/src/ohos_sdk/openharmony/native/llvm/bin/clang++)
set(CMAKE_CXX_FLAGS "--target=aarch64-linux-ohos")

#

# ohos

include_directories(${ELE_PATH}/src/)
include_directories(${ELE_PATH}/src/third_party/electron_node/deps/v8/include/)

```

```

include_directories(${ELE_PATH}/src/ohos_sdk/)
include_directories(${CMAKE_CURRENT_SOURCE_DIR}/../adapter/)

message(${CMAKE_CURRENT_SOURCE_DIR})

set(src demo.cc)

#

#add_executable(${TARGETFILENAME} demo.cpp)

#

#ADD_LIBRARY(${TARGETFILENAME} STATIC demo.cpp)

#
ADD_LIBRARY (${TARGETFILENAME} SHARED
    ${src}
)

#
# -ohos
target_link_libraries(${TARGETFILENAME} PUBLIC ${ELE_PATH}/src/out/musl_64/libelectron.so)
target_link_libraries(${TARGETFILENAME} PUBLIC ${CMAKE_CURRENT_SOURCE_DIR}/../adapter/build/libadaptertest.so)
target_compile_definitions(${TARGETFILENAME} PUBLIC NODE_MODULE_VERSION=116)

#
# .node
SET_TARGET_PROPERTIES(${TARGETFILENAME} PROPERTIES SUFFIX ".node")

```

4

addon.node

```

mkdir build
cd build
cmake ../
make

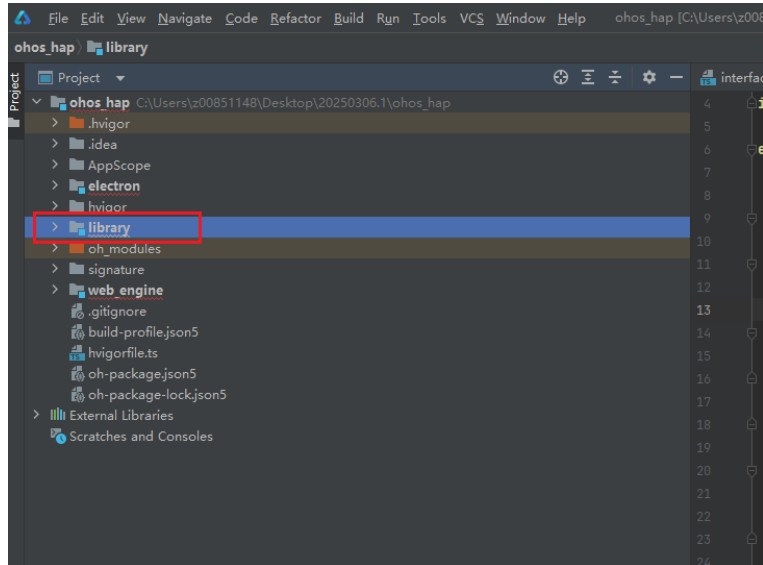
```

ets

har

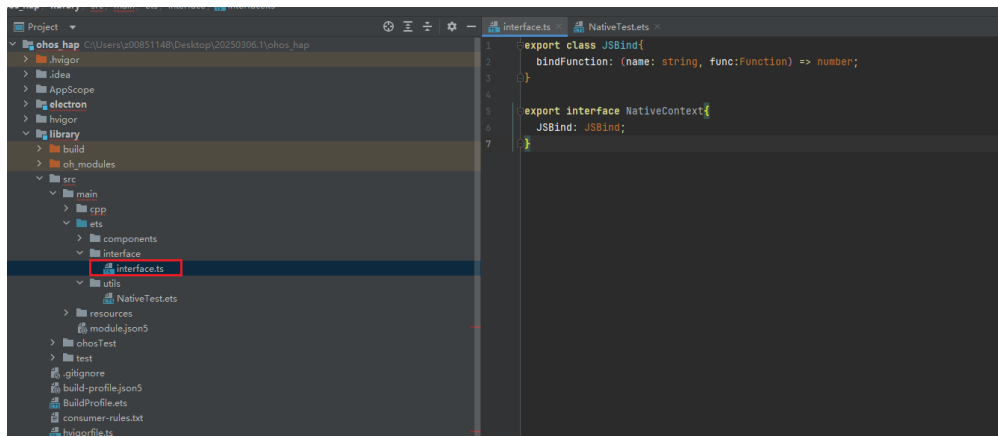
1

deveco File ==> New ==> Module ==> Static Library Library



2 adapter

2.1 adapter



```

export class JSBind{
  bindFunction: (name: string, func:Function) => number;
}

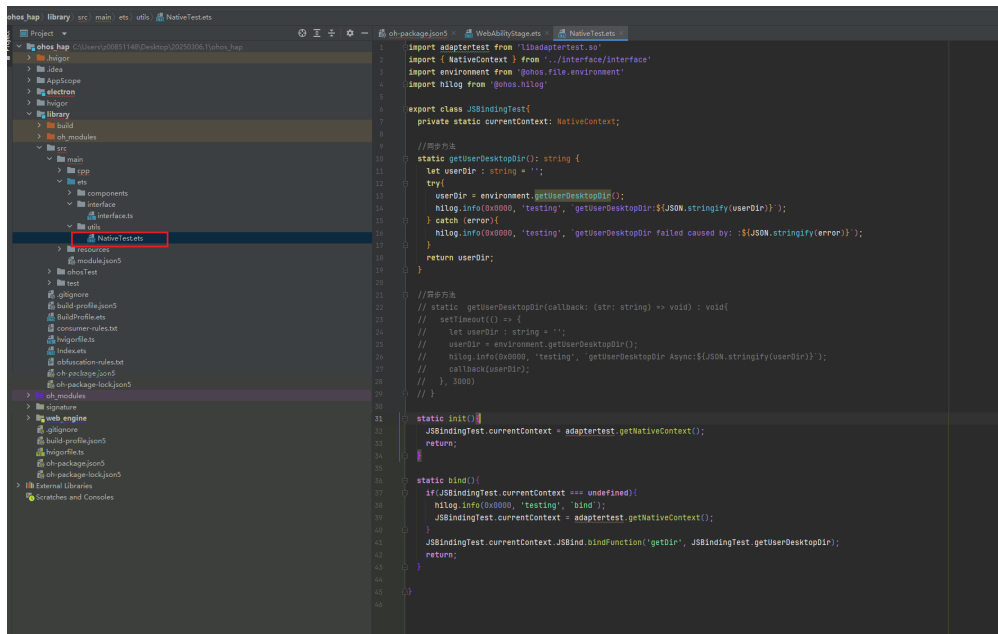
```

```

export interface NativeContext{
  JSBind: JSBind;
}

```

2.2 bind



```

import adaptestest from 'libadaptestest.so'
import { NativeContext } from '../interface/interface'
import environment from '@ohos.file.environment'
import hilog from '@ohos.hilog'

```

```

export class JSBindingTest{
  private static currentContext: NativeContext;

  //
  static getUserDesktopDir(): string {
    let userDir : string = '';
    try{
      userDir = environment.getUserDesktopDir();
      hilog.info(0x0000, 'testing', `getUserDesktopDir:${JSON.stringify(userDir)}`);
    } catch (error){
      hilog.info(0x0000, 'testing', `getUserDesktopDir failed caused by: ${JSON.stringify(error)}`);
    }
    return userDir;
  }

  //
  // static getUserDesktopDir(callback: (str: string) => void) : void{
  //   setTimeout(() => {
  //     let userDir : string = '';
  //     userDir = environment.getUserDesktopDir();
  //     hilog.info(0x0000, 'testing', `getUserDesktopDir Async:${JSON.stringify(userDir)}`);
  //     callback(userDir);
  //   }, 3000)
  // }

  static init(){
    JSBindingTest.currentContext = adaptestest.getNativeContext();
    return;
  }

  static bind(){
    if(JSBindingTest.currentContext === undefined){
      hilog.info(0x0000, 'testing', 'bind');
      JSBindingTest.currentContext = adaptestest.getNativeContext();
    }
  }
}

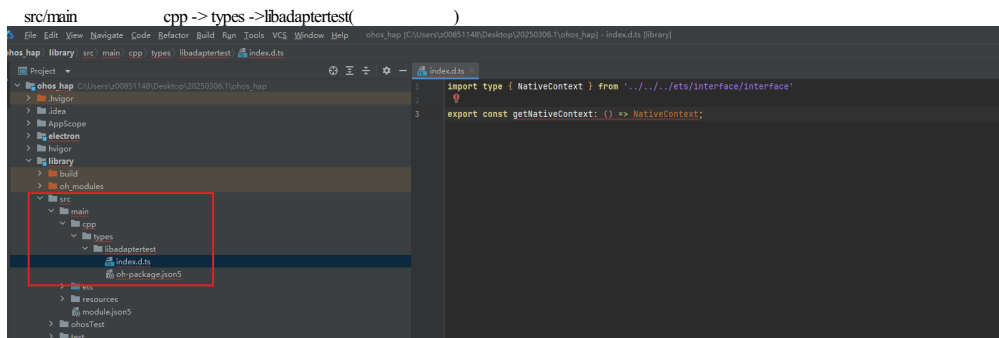
```

```

    }
    JSBindingTest.currentContext.JSBind.bindFunction('getDir', JSBindingTest.getUserDesktopDir);
    return;
  }
}

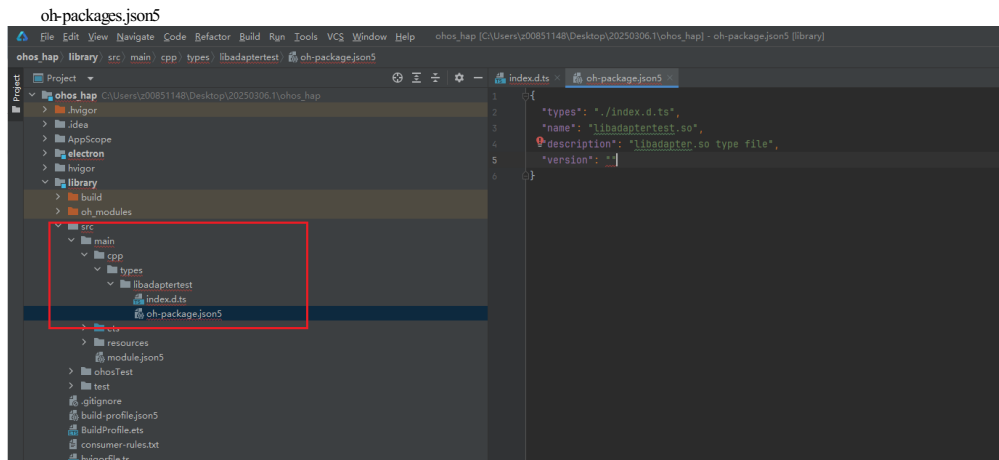
```

2.2.3



```
import type { NativeContext } from '../././ets/interface/interface'
```

```
export const getNativeContext: () => NativeContext;
```

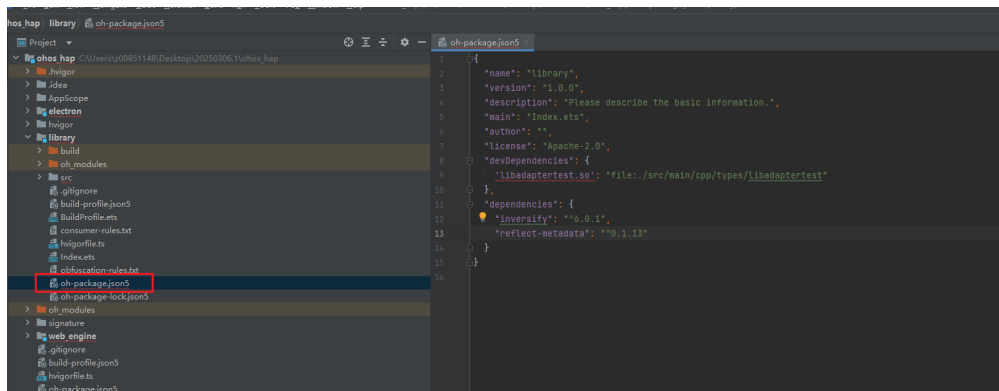


```

{
  "types": "./index.d.ts",
  "name": "libadaptestest.so",
  "description": "libadapter.so type file",
  "version": ""
}

```

2.2.4 library oh-packages.json5

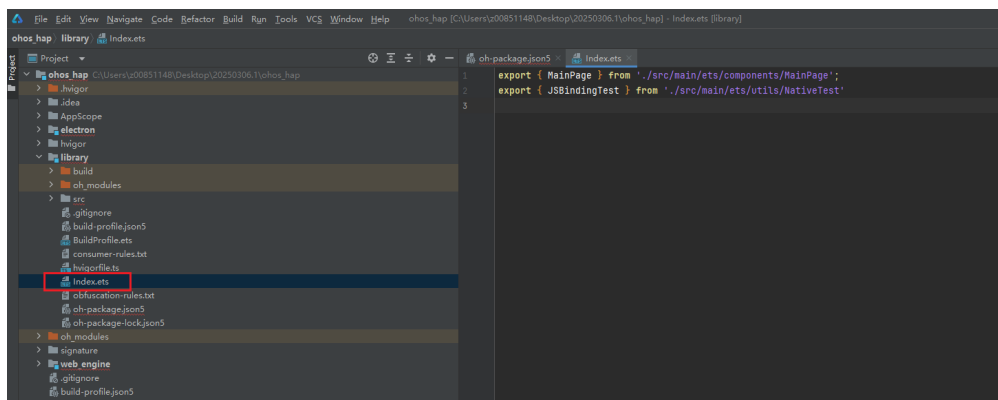


```

{
  "name": "library",
  "version": "1.0.0",
  "description": "Please describe the basic information.",
  "main": "Index.ets",
  "author": "",
  "license": "Apache-2.0",
  "devDependencies": {
    'libadaptestest.so': "file:./src/main/cpp/types/libadaptestest"
  },
  "dependencies": {
    "inversify": "^6.0.1",
    "reflect-metadata": "^0.1.13"
  }
}

```

2.2.5 library Index.ets

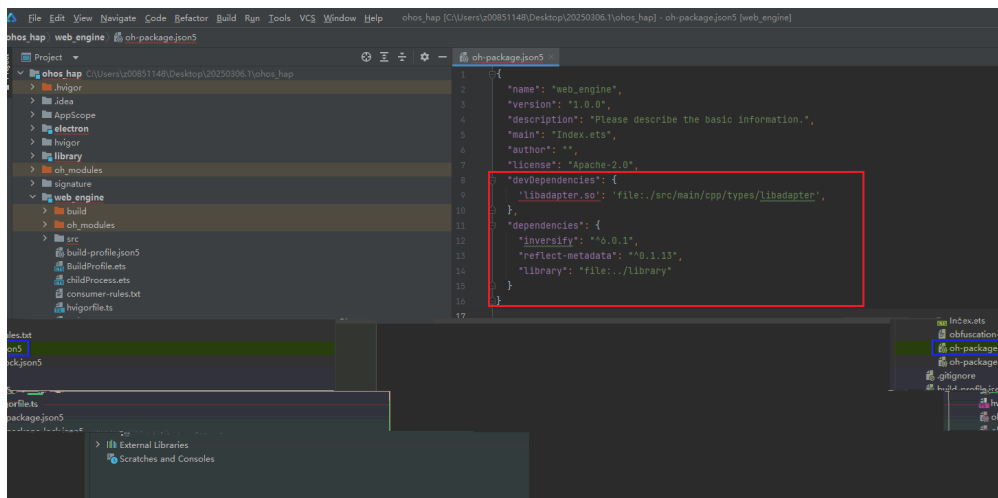


```
export { MainPage } from './src/main/ets/components/MainPage';
export { JSBindingTest } from './src/main/ets/Utils/NativeTest';
```

2.2.6 har

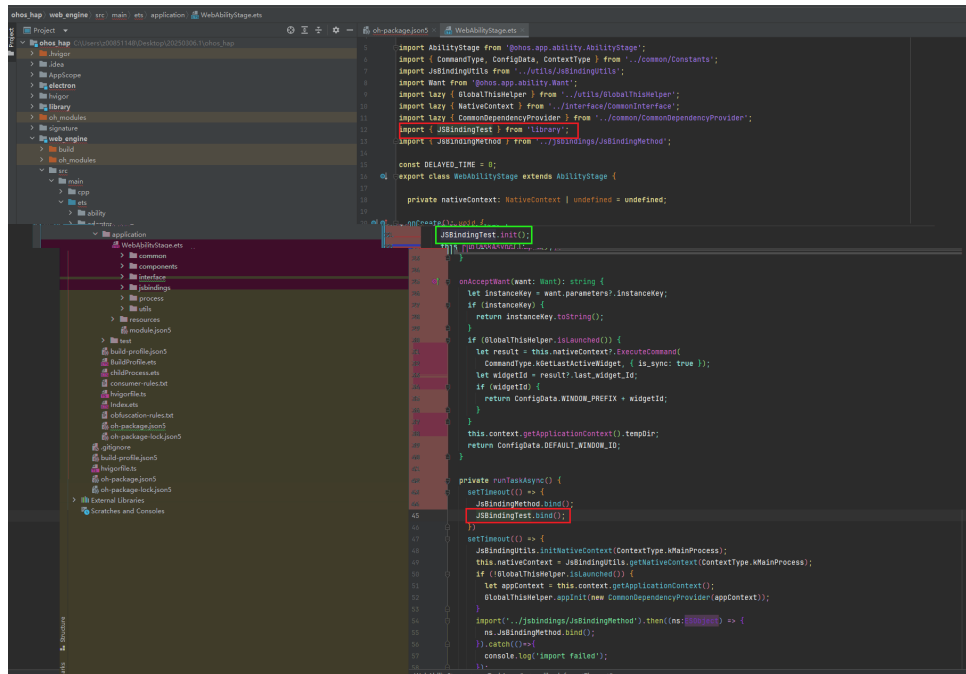
- / -DevEco Studio - HarmonyOS

1. web_engine HAR webengine/oh-packages.json5



```
{
  "name": "web_engine",
  "version": "1.0.0",
  "description": "Please describe the basic information.",
  "main": "Index.ets",
  "author": "",
  "license": "Apache-2.0",
  "devDependencies": {
    "libadapter.so": "file:./src/main/cpp/types/libadapter",
  },
  "dependencies": {
    "inversify": "^6.0.1",
    "reflect-metadata": "^0.1.13",
    "library": "file:../library"
  }
}
```

2. WebAbilityStage.ets



// Copyright (c) 2024 Huawei Device Co., Ltd. All rights reserved.
 // Use of this source code is governed by a BSD-style license that can be
 // found in the LICENSE file.

```
import AbilityStage from '@ohos.app.ability.AbilityStage';
import { CommandType, ConfigData, ContextType } from '../common/Constants';
import JsBindingUtils from '../utils/JsBindingUtils';
import Want from '@ohos.app.ability.Want';
import lazy { GlobalThisHelper } from '../utils/GlobalThisHelper';
import lazy { NativeContext } from '../interface/CommonInterface';
import lazy { CommonDependencyProvider } from '../common/CommonDependencyProvider';
import { JsBindingTest } from 'library';
import { JsBindingMethod } from '../jsbindings/JsBindingMethod';

const DELAYED_TIME = 0;
export class WebAbilityStage extends AbilityStage {

  private nativeContext: NativeContext | undefined = undefined;

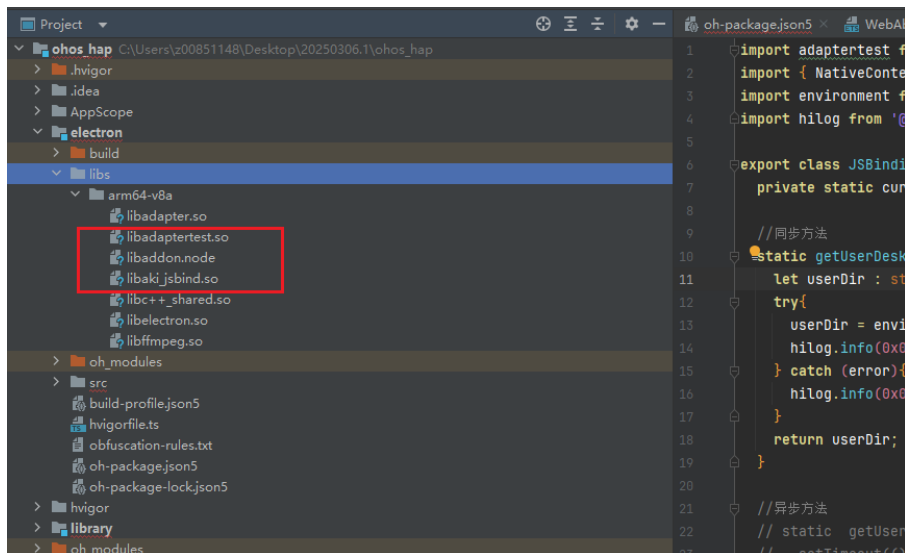
  onCreate(): void {
    JsBindingTest.init();
    this.runTaskAsync();
  }

  onAcceptWant(want: Want): string {
    let instanceKey = want.parameters?.instanceKey;
    if (instanceKey) {
      return instanceKey.toString();
    }
    if (GlobalThisHelper.isLaunched()) {
      let result = this.nativeContext?.ExecuteCommand(
        CommandType.kGetLastActiveWidget, { is_sync: true });
      let widgetId = result?.last_widget_Id;
      if (widgetId) {
        return ConfigData.WINDOW_PREFIX + widgetId;
      }
    }
    this.context.getApplicationContext().tempDir;
    return ConfigData.DEFAULT_WINDOW_ID;
  }

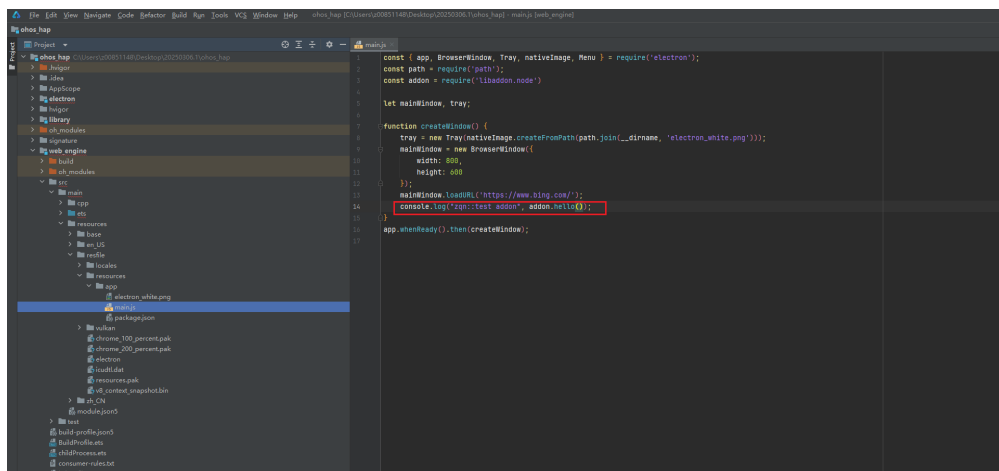
  private runTaskAsync() {
    setTimeout(() => {
      JsBindingMethod.bind();
      JsBindingTest.bind();
    })
    setTimeout(() => {
      JsBindingUtils.initNativeContext(ContextType.kMainProcess);
      this.nativeContext = JsBindingUtils.getNativeContext(ContextType.kMainProcess);
      if (!GlobalThisHelper.isLaunched()) {
        let appContext = this.context.getApplicationContext();
        GlobalThisHelper.appInit(new CommonDependencyProvider(appContext));
      }
      export { JsBindingMethod } from '../jsbindings/JsBindingMethod';
      no JsBindingMethod.bind();
    }).catch(() => {
      console.log('import failed');
    });
  }
}
```


2.2.7

- ## 1 HAP



2 JS



3

