```
const REQUEST_HEADERS = {
    'User-Agent':
      'Mozilla/5.0 (Macintosh; Intel Mac OS X
10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/94.0.4606.61 Safari/537.36',
    'Accept-Language': 'en',
}

// 即将登陆
const STATUS_COMING = 2
// 支持解锁
const STATUS_AVAILABLE = 1
// 不支持解锁
const STATUS_NOT_AVAILABLE = 0
// 检测超时
const STATUS_TIMEOUT = -1
// 检测异常
const STATUS_ERROR = -2

function getFlagEmoji(code) {
  const codePoints = code
     .toUpperCase()
    .split('')
    .map((char) => 127397 + char.charCodeAt());
  return String.fromCodePoint(...codePoints);
}
function getFlagEmoji(region) {
  const codePoints = region
     .toUpperCase()
    .split('')
    .map((char) => 127397 + char.charCodeAt());
  return String.fromCodePoint(...codePoints);
}

const UA = 'Mozilla/5.0 (Macintosh; Intel Mac OS X
10_15_7) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/94.0.4606.61 Safari/537.36'


  ;(async () => {
    let panel_result = {
      title: '📺 流媒体解锁检测 🔍',
      content: '',
      icon: '4k.tv.fill',
      'icon-color': '#FF2D55',
    }
  let [{ region, status }] = await
Promise.all([testDisneyPlus()])
    await
Promise.all([check_youtube_premium(),check_netflix()
])
      .then((result) => {
         console.log(result)
 let disney_result=""
    if (status==STATUS_COMING) {
        //console.log(1)
        disney_result="Disney+: 即将登陆
~"+region.toUpperCase()
      } else if (status==STATUS_AVAILABLE){
        //console.log(2)
        console.log(region)
        disney_result="Disney+: 已解锁 ➾ "
+`${getFlagEmoji(region)} | ` + region.toUpperCase()
        // console.log(result["Disney"])
      } else if (status==STATUS_NOT_AVAILABLE) {
        //console.log(3)
        disney_result="Disney+: 未支持 🚫 "
      } else if (status==STATUS_TIMEOUT) {
        disney_result="Disney+: 检测超时 🚦"
      }
result.push(disney_result)
console.log(result)
        let content = result.join('\n')
        console.log(content)

panel_result['content'] = content
      })
      .finally(() => {
        $done(panel_result)
      })
  })()
  async function check_youtube_premium() {
    let inner_check = () => {
      return new Promise((resolve, reject) => {
        let option = {
          url: 'https://www.youtube.com/premium',
          headers: REQUEST_HEADERS,
        }
        $httpClient.get(option, function (error,
response, data) {
          if (error != null || response.status !==
200) {
            reject('Error')
            return
          }

          if (data.indexOf('Premium is not available
in your country') !== -1) {
            resolve('Not Available')
            return
          }

          let region = ''
          let re = new RegExp('"countryCode":"
(.*?)"', 'gm')
          let result = re.exec(data)
          if (result != null && result.length === 2)
{
            region = result[1]
          } else if (data.indexOf('www.google.cn')
!== -1) {
            region = 'CN'
          } else {
            region = 'US'
          }
          resolve(region)
        })
      })
    }

    let youtube_check_result = 'YouTube: '

    await inner_check()
      .then((code) => {
        if (code === 'Not Available') {
          youtube_check_result += '不支持解锁🚫'
        } else {
          youtube_check_result += '已解锁 ➾ '
+`${getFlagEmoji(code)} | `+ code.toUpperCase()
        }
      })
      .catch((error) => {
        youtube_check_result += '检测失败，请刷新面板
🔄'
      })

    return youtube_check_result
  }

  async function check_netflix() {
    let inner_check = (filmId) => {
      return new Promise((resolve, reject) => {
        let option = {
          url: 'https://www.netflix.com/title/' +
filmId,
          headers: REQUEST_HEADERS,
        }
        $httpClient.get(option, function (error,
response, data) {
          if (error != null) {
            reject('Error')
            return
          }

          if (response.status === 403) {
            reject('Not Available')
            return
          }

          if (response.status === 404) {
            resolve('Not Found')
            return
          }

          if (response.status === 200) {
            let url = response.headers['x-
originating-url']
            let region = url.split('/')[3]
            region = region.split('-')[0]
            if (region == 'title') {
              region = 'us'
            }
            resolve(region)
            return
          }

          reject('Error')
        })
      })
    }

    let netflix_check_result = 'Netflix: '

    await inner_check(81215567)
      .then((code) => {
        if (code === 'Not Found') {
          return inner_check(80018499)
        }
        netflix_check_result += '已完整解锁 ➾ '
+`${getFlagEmoji(code)} | ` + code.toUpperCase()
        return Promise.reject('BreakSignal')
      })
      .then((code) => {
        if (code === 'Not Found') {
          return Promise.reject('Not Available')
        }

        netflix_check_result += '仅自制剧，区域： ' +
`${getFlagEmoji(code)} | ` + code.toUpperCase()
        return Promise.reject('BreakSignal')
      })
      .catch((error) => {
        if (error === 'BreakSignal') {
          return
        }
        if (error === 'Not Available') {
          netflix_check_result += '该节点不支持解锁🚫'
          return
        }
        netflix_check_result += '检测失败，请刷新面板
🔄'
      })

    return netflix_check_result
  }

  async function testDisneyPlus() {
    try {
        let { region, cnbl } = await
Promise.race([testHomePage(), timeout(7000)])
        console.log(`homepage: region=${region},
cnbl=${cnbl}`)
        // 即将登陆
    //  if (cnbl == 2) {
    //    return { region, status: STATUS_COMING }
    //  }
        let { countryCode, inSupportedLocation } =
await Promise.race([getLocationInfo(),
timeout(7000)])
        console.log(`getLocationInfo:
countryCode=${countryCode},
inSupportedLocation=${inSupportedLocation}`)

        region = countryCode ?? region
        console.log( "region:"+region)
        // 即将登陆
        if (inSupportedLocation === false ||
inSupportedLocation === 'false') {
          return { region, status: STATUS_COMING }
        } else {
          // 支持解锁
          return { region, status: STATUS_AVAILABLE
}
        }

      } catch (error) {
        console.log("error:"+error)

        // 不支持解锁
        if (error === 'Not Available') {
          console.log("不支持")
          return { status: STATUS_NOT_AVAILABLE }
        }

        // 检测超时
        if (error === 'Timeout') {
          return { status: STATUS_TIMEOUT }
        }

        return { status: STATUS_ERROR }
      }

    }

      function getLocationInfo() {
        return new Promise((resolve, reject) => {
          let opts = {
            url:
'https://disney.api.edge.bamgrid.com/graph/v1/device
/graphql',
            headers: {
              'Accept-Language': 'en',
              Authorization:
'ZGlzbmV5JmJyb3dzZXImMS4wLjA.Cu56AgSfBTDag5NiRA81oLH
kDZfu5L3CKadnefEAY84',
              'Content-Type': 'application/json',
              'User-Agent': UA,
            },
            body: JSON.stringify({
              query: 'mutation
registerDevice($input: RegisterDeviceInput!) {
registerDevice(registerDevice: $input) { grant {
grantType assertion } } }',
              variables: {
                input: {
                  applicationRuntime: 'chrome',
                  attributes: {
                    browserName: 'chrome',
                    browserVersion: '94.0.4606',
                    manufacturer: 'apple',
                    model: null,
                    operatingSystem: 'macintosh',
                    operatingSystemVersion:
'10.15.7',
                    osDeviceIds: [],
                  },
                  deviceFamily: 'browser',
                  deviceLanguage: 'en',
                  deviceProfile: 'macosx',
                },
              },
            }),
          }

          $httpClient.post(opts, function (error,
response, data) {
            if (error) {
              reject('Error')
              return
            }

            if (response.status !== 200) {
              console.log('getLocationInfo: ' +
data)
              reject('Not Available')
              return
            }

            data = JSON.parse(data)
            if(data?.errors){
              console.log('getLocationInfo: ' +
data)
              reject('Not Available')
              return
            }

            let {
              token: { accessToken },
              session: {
                inSupportedLocation,
                location: { countryCode },
              },
            } = data?.extensions?.sdk
            resolve({ inSupportedLocation,
countryCode, accessToken })
          })
        })
      }

      function testHomePage() {
        return new Promise((resolve, reject) => {
          let opts = {
            url: 'https://www.disneyplus.com/',
            headers: {
              'Accept-Language': 'en',
              'User-Agent': UA,
            },
          }

          $httpClient.get(opts, function (error,
response, data) {
            if (error) {
              reject('Error')
              return
            }
            if (response.status !== 200 ||
data.indexOf('unavailable') !== -1) {
              reject('Not Available')
              return
            }

            let match = data.match(/Region: ([A-Za-
z]{2})[\s\S]*?CNBL: ([12])/)
            if (!match) {
              resolve({ region: '', cnbl: '' })
              return
            }

            let region = match[1]
            let cnbl = match[2]
            resolve({ region, cnbl })
          })
        })
      }

      function timeout(delay = 5000) {
        return new Promise((resolve, reject) => {
          setTimeout(() => {
            reject('Timeout')
          }, delay)
        })
      }
```