font-weight: bold;

color: #FFFFFF;

}

.btn-qrcode {

width: 48%;

height: 88rpx;

line-height: 88rpx;

border-radius: 8rpx;

font-size: 36rpx;

font-weight: bold;

color: #000000;

border: 1px solid #999999;

}

}

}

</style>

<template>

<view class="body">

<!--Status bar, -->

<view class="status\_bar" style="width: 100%;background: #f8f8f8;" :style="{height:topHeight+'px'}"></view>

<view class="nav">

<view class="nav-left">

<uni-icons class="icon" type="back" size="24" @click="onClickBack"></uni-icons>

<text>{{selectToken.name}}</text>

</view>

</view>

<view class="top">

<view class="top-content">

<view class="blance">

{{AMOUNT}} {{selectToken.name}}

</view>

<view class="price">

~￥{{showPrice}}

</view>

<view class="status">

<image src="../../static/icon\_selected.png"></image>

<text></text>

</view>

</view>

</view>

<view class="content">

<view class="row">

<view class="row-name">

</view>

<view class="row-right">

{{showDate}}

</view>

</view>

<view class="row">

<view class="row-name">

</view>

<view class="row-right">

<text>{{showAddress(TOADDRESS)}}</text>

<image src="../../static/img\_copy.png" @click="onClickCopyAddress"></image>

</view>

</view>

<view class="row">

<view class="row-name">

</view>

<view class="row-right">

<text>{{showAddress(response.TXID)}}</text>

<image src="../../static/img\_copy.png" @click="onClickTXID"></image>

</view>

</view>

<view class="row">

<view class="row-name">

</view>

<view class="row-right">

{{$store.state.chainList.find(item=>item.symbol==getSelectChain()).name}}

</view>

</view>

</view>

</view>

</template>

<script>

export default {

data() {

return {

topHeight: 0,

selectToken: {},

response: {

TXID: ''

},

USD\_PRICE: 0,

TOADDRESS: '',

AMOUNT: 0,

TOKEN\_PRICE: 0,

showPrice: 0,

showDate: null

}

},

onLoad(option) {

let topHeight = uni.getSystemInfoSync().statusBarHeight

this.topHeight = topHeight

this.selectToken = JSON.parse(option.selectToken)

this.response = JSON.parse(option.response)

this.USD\_PRICE = option.USD\_PRICE

this.TOADDRESS = option.TOADDRESS

this.AMOUNT = option.AMOUNT

this.TOKEN\_PRICE = option.TOKEN\_PRICE

this.showPrice = this.toFixed(parseFloat(this.TOKEN\_PRICE) \* parseFloat(this.USD\_PRICE) \* parseFloat(this

.AMOUNT), 6)

this.showDate = this.dateTimeStr('y-m-d h:i')

},

methods: {

onClickBack() {

uni.navigateBack({

delta: 1

})

},

onClickCopyAddress() {

uni.setClipboardData({

data: this.TOADDRESS,

success: function() {

uni.showToast({

title: ,

icon: 'none'

})

}

});

},

onClickCopyTXID() {

uni.setClipboardData({

data: this.response.TXID,

success: function() {

uni.showToast({

title:

icon: 'none'

})

}

});

},

dateTimeStr(str) {

var date = new Date(),

year = date.getFullYear(), / / year

month = date.getMonth() + 1, / / month

day = date.getDate(), / / day

hour = date.getHours() < 10 ?"0" + date.getHours() : date.getHours(), / / time

minute = date.getMinutes() < 10 ?date.getMinutes() : date.getMinutes(), / / points

second = date.getSeconds() < 10 ?date.getSeconds() : date.getSeconds(); / / sec

month >= 1 && month <= 9 ?(month = "0" + month) : "";

day >= 0 && day <= 9 ?(day = "0" + day) : "";

hour >= 0 && hour <= 9 ?hour : "";

minute >= 0 && minute <= 9 ?(minute = "0" + minute) : "";

second >= 0 && second <= 9 ?(second = "0" + second) : "";

if (str.indexOf('y') != -1) {

str = str.replace('y', year)

}

if (str.indexOf('m') != -1) {

str = str.replace('m', month)

}

if (str.indexOf('d') != -1) {

str = str.replace('d', day)

}

if (str.indexOf('h') != -1) {

str = str.replace('h', hour)

}

if (str.indexOf('i') != -1) {

str = str.replace('i', minute)

}

if (str.indexOf('s') != -1) {

str = str.replace('s', second)

}

return str;

}

}

}

</script>

<style lang="scss" scoped>

page {

height: 100%;

}

.body {

height: 100%;

background: #F8F8F8;

}

.nav {

padding: 24rpx 26rpx;

.nav-left {

text-align: center;

position: relative;

.icon {

position: absolute;

left: 0;

}

text {

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

}

}

.top {

padding: 0 26rpx;

.top-left {

margin: 20rpx 0;

display: flex;

align-items: center;

justify-content: center;

.left-icon {

margin-right: auto;

}

text {

margin-right: auto;

font-size: 36rpx;

font-weight: bolder;

color: #000000;

}

}

.top-content {

display: flex;

flex-direction: column;

align-items: center;

margin-top: 200rpx;

.blance {

font-size: 48rpx;

font-weight: bolder;

color: #000000;

}

.price {

margin-top: 26rpx;

font-size: 26rpx;

font-weight: 400;

color: #999999;

}

.status {

margin-top: 26rpx;

margin-bottom: 66rpx;

display: flex;

align-items: center;

image {

width: 40rpx;

height: 40rpx;

margin-right: 12rpx;

}

text {

font-size: 26rpx;

font-weight: 400;

color: #6C2AEE;

}

}

}

}

.content {

background: #FFFFFF;

padding: 40rpx 30rpx 0 30rpx;

display: flex;

flex-direction: column;

.row {

display: flex;

align-items: center;

margin-bottom: 42rpx;

.row-name {

font-size: 28rpx;

color: #999999;

}

.row-right {

margin-left: auto;

font-size: 28rpx;

color: #000000;

display: flex;

align-items: center;

image {

width: 26rpx;

height: 30rpx;

margin-left: 18rpx;

}

}

}

}

</style>

<template>

<view>

<!-- -->

<view class="status\_bar" style="width: 100%;background: #f8f8f8;" :style="{height:topHeight+'px'}"></view>

<view class="content">

<view class="nav">

<view class="nav-left">

<uni-icons class="icon" type="back" size="24" @click="onClickBack"></uni-icons>

<text></text>

</view>

</view>

<!--<view class="top">

<view class="top-left">

<uni-icons type="back" size="24" @click="onClickBack"></uni-icons>

<text></text>

</view>

<!-- <view class="top-right">

<image class="logo" src="../../static/img\_pyy\_logo.png"></image>

PYY

<uni-icons type="bottom"></uni-icons>

</view> -->

<!-- <view class="search">

<image class="search-icon" src="../../static/icon\_search\_input.png"></image>

<input class="search-input" type="text" placeholder=

placeholder-style="font-size:26rpx;color:#999999;">

</view> -->

<view class="bottom">

<view class="title">

</view>

<view class="token-list">

<view class="token-item" v-for="(item,index) in tokenList" :key="index"

@click="onClickTokenItem(index)">

<image class="token-icon" :src="item.iconUrl">

</image>

<view class="token-info">

<text class="token-name">{{item.name}}</text>

<text class="token-price">{{item.address}}</text>

</view>

<!-- <view class="token-number">

<text class="token-count">{{item.number.toFixed(2)}}</text>

<text class="token-total">￥ {{item.total.toFixed(2)}}</text>

</view> -->

<image class="icon-select" v-show="item.select" src="../../static/icon\_selected.png"></image>

<image class="icon-select" v-show="!item.select" src="../../static/icon\_select.png"></image>

</view>

</view>

</view>

</view>

<uni-popup ref="alertDialog" type="dialog">

<uni-popup-dialog: type= "msgType" cancelText= "Cancel" confirmText= "confirm delete" title= "Action prompt" content= " confirm delete display?"

@confirm="dialogConfirm" @close="dialogClose"></uni-popup-dialog>

</uni-popup>

</view>

</template>

<script>

import {

getTokens

} from "@/api/wallet.js"

export default {

data() {

return {

topHeight: 0,

searchValue: "",

tokenList: [],

msgType: 'info',

selectIndex: 0

};

},

onLoad() {

let topHeight = uni.getSystemInfoSync().statusBarHeight

this.topHeight = topHeight

},

async onShow() {

var that = this

let newTokenList = []

await getTokens(this.getSelectChain()).then(res => {

if (res.STATUS == 200) {

// let mainToken = {

name: res.DATA.MAINCOIN,

address: that.$store.state.chainList.find(item => item.symbol === that

.getSelectChain()).note,

iconUrl: that.$store.state.chainList.find(item => item.symbol === that

.getSelectChain()).logoUrl,

select: that.getSelectWallet().TOKENLIST.includes(res.DATA.MAINCOIN)

}

newTokenList.push(mainToken)

// console.log(that.getSelectWallet());

for (let key in res.DATA.TOKENS) {

let token = {

name: res.DATA.TOKENS[key].CONTRACT\_NAME,

address: that.showAddress(res.DATA.TOKENS[key].CONTRACT\_ADDRESS),

iconUrl: res.DATA.TOKENS[key].CONTRACT\_ICON,

select: that.getSelectWallet().TOKENLIST.includes(key.toLocaleUpperCase())

}

newTokenList.push(token)

}

that.tokenList = newTokenList

} else {

uni.showToast({

title: '' + res.MESSAGE,

duration: 3000,

icon: "none"

})

that.tokenList = []

}

}).catch(err => {

uni.showToast({

title: ':' + err,

duration: 3000,

icon: "none"

})

that.tokenList = []

})

},

methods: {

onClickBack() {

uni.navigateBack({

delta: 1

})

},

onClickTokenItem(index) {

this.selectIndex = index;

if (this.tokenList[this.selectIndex].select) {

this.$refs.alertDialog.open()

} else {

let wallet = this.getSelectWallet()

if (!wallet.TOKENLIST.includes(this.tokenList[this.selectIndex].name.toLocaleUpperCase()))

wallet.TOKENLIST.push(this.tokenList[this.selectIndex].name.toLocaleUpperCase())

this.setSelectWallet(wallet)

this.saveWallet(wallet)

this.tokenList[this.selectIndex].select = !this.tokenList[this.selectIndex].select

console.log(this.getWalletList());

}

},

dialogConfirm() {

let wallet = this.getSelectWallet()

wallet.TOKENLIST = wallet.TOKENLIST.filter(item => item != this.tokenList[this.selectIndex].name.toLocaleUpperCase())

this.setSelectWallet(wallet)

this.saveWallet(wallet)

this.tokenList[this.selectIndex].select = !this.tokenList[this.selectIndex].select

this.$refs.alertDialog.close()

},

dialogClose() {

this.$refs.alertDialog.close()

}

}

}

</script>

<style lang="scss">

</style>

<style lang="scss" scoped>

/deep/ .uni-dialog-title-text {

font-weight: bold !important;

color: #000 !important;

}

/deep/ .uni-button-color {

color: #6C2AEE !important;

}

.content {

background: #F8F8F8;

.nav {

padding: 24rpx 26rpx;

.nav-left {

text-align: center;

position: relative;

.icon {

position: absolute;

left: 0;

}

text {

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

}

}

.top {

padding: 0 26rpx;

display: flex;

align-items: center;

.top-left {

margin: 20rpx 0;

display: flex;

align-items: center;

text {

font-size: 36rpx;

font-weight: bolder;

color: #000000;

}

}

.top-right {

margin-top: 16rpx;

margin-left: auto;

width: 240rpx;

height: 80rpx;

background: #F8F8F8;

border: 1px solid #EBEBEB;

border-radius: 40rpx;

display: flex;

align-items: center;

font-size: 30rpx;

font-weight: bold;

color: #000000;

.logo {

margin-left: 30rpx;

margin-right: 30rpx;

width: 40rpx;

height: 40rpx;

}

.uni-icons {

margin-left: auto;

margin-right: 20rpx;

}

}

}

.search {

padding: 0 30rpx;

margin-top: 30rpx;

padding-bottom: 30rpx;

position: relative;

.search-icon {

width: 30rpx;

height: 30rpx;

position: absolute;

left: 52rpx;

transform: translateY(80%);

}

.search-input {

height: 70rpx;

background: #FFFFFF;

border: 1px solid #EBEBEB;

border-radius: 35rpx;

padding-left: 62rpx;

font-size: 26rpx;

}

}

.bottom {

background-color: #FFFFFF;

padding: 0 30rpx;

padding-top: 30rpx;

.title {

font-size: 30rpx;

font-weight: bold;

color: #000000;

}

.token-list {

display: flex;

flex-direction: column;

margin-top: 46rpx;

.token-item {

display: flex;

align-items: center;

margin-bottom: 46rpx;

.token-icon {

width: 80rpx;

height: 80rpx;

margin-right: 30rpx;

}

.token-info {

display: flex;

flex-direction: column;

.token-name {

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

.token-price {

font-size: 24rpx;

font-weight: 400;

color: #999999;

}

}

.token-number {

margin-left: auto;

display: flex;

flex-direction: column;

text-align: right;

.token-count {

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

.token-total {

font-size: 24rpx;

font-weight: 400;

color: #999999;

}

}

.icon-select {

margin-left: auto;

width: 40rpx;

height: 40rpx;

}

}

}

}

}

</style>

<template>

<view>

<!--Status bar, -->

<view class="status\_bar" :style="{height:topHeight+'px'}"></view>

<!--Navigation bar, -->

<uni-nav-bar left-icon="left" @clickLeft="onClickGoBack" :border="false">

<view class="nav-title">

<view class="title-content">

<text></text>

<view class="title-bottom">

<view class="bottom-split" :class="{'split-active':step===1}"></view>

<view class="bottom-split" :class="{'split-active':step===2}"></view>

</view>

</view>

</view>

</uni-nav-bar>

<!--Page body-->

<view class="content">

<view class="body-title"></view>

<view class="body-msg">

</view>

<view class="body-msg-tip">

<uni-icons v-show="!isShowPwd" type="eye-slash" size="20" @click="onClickEye"></uni-icons>

<uni-icons v-show="isShowPwd" type="eye" size="20" @click="onClickEye"></uni-icons>

</view>

<input class="cinput" adjust-position="false" auto-blur="true" @blur="blur" @input="codenum" :focus="focus"

value="pwd" v-model="pwd" type="tel" maxlength="6" :cursor="pwd.length" />

<view class="code-input">

<view v-for="(item,index) in 6" :key="index" @click="codefocus(index)"

:style='(index == pwd.length?"border: 2rpx solid #7729FB;":"color: " + codeColor + ";" +"border: 2rpx solid #EBEBEB")'>

{{showPassword(index)}}

</view>

</view>

</view>

</view>

</template>

<script>

export default {

data() {

return {

title: 'Hello',

topHeight: 0,

step: 1,

Fus: true, / / input focus for keyboard hiding and reocation

codeColor: "# 313131", / / Color of the custom cursor

Pwd: '', / / This is the password entered by the user,

isShowPwd: False, / / whether the password is displayed

Type: "create" / / first load type is created by default

}

},

onLoad(option) {

let topHeight = uni.getSystemInfoSync().statusBarHeight

this.topHeight = topHeight

this.type = option.type

},

methods: {

onClickGoBack() {

uni.navigateBack();

},

onClickEye() {

this.isShowPwd = !this.isShowPwd

},

showPassword(index) {

if (this.pwd[index] === undefined)

return ''

return this.isShowPwd ?this.pwd[index] : "●"

},

// enter password

codenum: function(event) {

console.log(event);

var that = this

var pwd = event.target.value.replace(/\D/g, '')

console.log(pwd);

that.pwd = pwd

if (pwd.length == 6) {

uni.navigateTo({

url: '../create/verify?pwd=' + this.pwd + '&type=' + this.type

})

this.pwd = ''

}

},

// The keyboard is hidden after the setting loses focus

blur: function() {

var that = this

that.focus = false

},

// Click on the custom cursor to display the keyboard

codefocus: function(e) {

var that = this

this.focus = true

}

}

}

</script>

<style lang="scss" scoped>

.status\_bar {

width: 100%;

}

.nav-title {

width: 100%;

display: flex;

align-items: center;

justify-content: center;

.title-content {

display: flex;

flex-direction: column;

align-items: center;

text {

text-align: center;

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

.title-bottom {

display: flex;

.bottom-split {

margin: 2px;

width: 60rpx;

height: 6rpx;

background: #E6E6E6;

border-radius: 2rpx;

}

.split-active {

background: #7729FB;

}

}

}

}

.content {

display: flex;

flex-direction: column;

padding-left: 80rpx;

.body-title {

margin-top: 90rpx;

font-size: 48rpx;

font-family: Source Han Sans CN;

font-weight: bold;

color: #000000;

}

.body-msg {

margin-top: 30rpx;

margin-right: 60rpx;

font-size: 28rpx;

font-weight: 400;

color: #000000;

}

.body-msg-tip {

margin-top: 100rpx;

font-size: 28rpx;

font-weight: 400;

color: #000000;

display: flex;

align-items: center;

.uni-icons {

margin-left: 20rpx;

}

}

}

.code-input {

display: flex;

}

.code-input>view {

margin-top: 30rpx;

margin-right: 16rpx;

width: 86rpx;

height: 86rpx;

line-height: 86rpx;

font-size: 48rpx;

font-weight: bold;

color: #313131;

text-align: center;

border-radius: 10rpx;

background: #F8F8F8;

}

.code-input>view:nth-child(1) {

margin-left: 0rpx;

}

.cinput {

position: fixed;

left: -100rpx;

width: 50rpx;

height: 50rpx;

}

</style>

<template>

<view>

<!--Status bar, -->

<view class="status-bar" :style="{height:topHeight+'px'}"></view>

<view class="nav">

<view class="nav-left">

<uni-icons class="icon" type="back" size="24" @click="onClickBack"></uni-icons>

<text></text>

<image class="icon-scan" @click="onClickScan" src="../../static/img\_scan.png"></image>

</view>

</view>

<view class="type">

<view class="type-item" :class="importType=='words'?'type-item-active':''" @click="onClickType('words')">

</view>

<view class="type-item" :class="importType=='prikey'?'type-item-active':''" @click="onClickType('prikey')">

</view>

</view>

<view class="input-container" v-show="importType=='words'">

<textarea class="type-input" v-model. Trim = "words" placeholder= "Please enter mnemonic words in order and use the space space"

placeholder-style="font-size:26rpx;color:#999999;">

</textarea>

<Text @ click = "onClickPaste ('words')"> Paste </ text>

</view>

<view class="input-container" v-show="importType=='prikey'">

<textarea class="type-input" v-model. Trim = "prikey" placeholder= "Please enter or paste a private key"

placeholder-style="font-size:26rpx;color:#999999;">

</textarea>

<Text @ click = "onClickPaste ('prikey'),"> Paste </ text>

</view>

<view class="name">

</view>

<input class="input-name" type="text" v-model.trim="walletName" placeholder=

placeholder-style="font-size:26rpx;color:#999999;" maxlength="6">

<!-- <view class="senior">

<text class= "senior-name"> Advanced Mode </ text>

<uni-icons type="right" color="#6C2AEE" size="18"></uni-icons>

</view> -->

<view class="bottom">

<button class= "btn-confirm" type= "primary" @click= "onClickSelectChain"> confirm import </ button>

</view>

<select-chain ref="chain" @importWallet="importWallet"></select-chain>

</view>

</template>

<script>

import {

createAddressByMnemonic,

createAddressByPkey

} from "@/api/wallet.js"

export default {

data() {

return {

topHeight: 0,

importType: 'words', / / import means word mnemonic prikey private key

words: '', //

prikey: '', //

walletName: '', //

The pwd: '' / / app password

}

},

onLoad(option) {

let topHeight = uni.getSystemInfoSync().statusBarHeight

this.topHeight = topHeight

this.pwd = option.pwd

console.log(option.pwd);

},

methods: {

onClickBack() {

uni.navigateBack({

delta: 1

})

},

onClickScan() {

var that = this

uni.scanCode({

success: function(res) {

console. Logs ('barcode type:' + res.scanType);

console. Logs ('barcode content:' + res.result);

if (res.result) {

if (that.importType === 'words')

that.words = res.result

if (that.importType === 'prikey')

that.prikey = res.result

}

}

});

},

onClickType(type) {

this.importType = type

},

onClickPaste(type) {

var that = this

uni.getClipboardData({

success: function(res) {

if (type === 'words') {

that.words = res.data

} else if (type === 'prikey') {

that.prikey = res.data

}

}

});

},

onClickSelectChain() {

if (!this.words && this.importType === 'words') {

uni.showToast({

title:

duration: 1500,

icon: "none"

})

return

} else if (!this.prikey && this.importType === "prikey") {

uni.showToast({

title:

duration: 1500,

icon: "none"

})

return

}

this.$refs.chain.openSelectChain()

},

// type:bng/eth/tron

importWallet(symbol) {

console.log( + symbol);

var that = this

let data = {

type: symbol,

value: this.importType === 'words' ?this.words : this.prikey

}

//

let walletList = this.getWalletList();

switch (symbol) {

case "BEP20":

if (walletList.BNB.find(item => item.PRIKEY === data.value || item.MNEMONIC === data

.value) != undefined) {

uni.showToast({

title:

duration: 3000,

icon: "none"

})

return

}

break;

case "ERC20":

if (walletList.ETH.find(item => item.PRIKEY === data.value || item

.MNEMONIC === data

.value) != undefined) {

uni.showToast({

title:

duration: 3000,

icon: "none"

})

return

}

break;

case "TRC20":

if (walletList.TRON.find(item => item.PRIKEY === data.value || item.MNEMONIC === data

.value) != undefined) {

uni.showToast({

title:

duration: 3000,

icon: "none"

})

return

}

break;

default:

break;

}

if (this.importType === 'words') {

createAddressByMnemonic(data).then(res => {

this.requestSuccess(res)

}).catch(err => {

uni.showToast({

title:

duration: 3000,

icon: "none"

})

})

} else {

createAddressByPkey(data).then(res => {

this.requestSuccess(res)

}).catch(err => {

uni.showToast({

title:

duration: 3000,

icon: "none"

})

})

}

},

requestSuccess(res) {

var that = this

if (res.STATUS === 200) {

//

let walletInfo = {

HEADLOGO: require("@/static/head/head\_photo\_" + Math.floor(Math.random() \* 10 + 1) + ".jpeg"),

PRIKEY: res.PRIKEY, //

MNEMONIC: res.MNEMONIC, //

SYMBOL: res.SYMBOL, //

ADDRESS: res.ADDRESS, //

NAME: that.walletName.length <= 0 ?'' : that.walletName, //

BACKUPS: true, //

TOKENLIST: res.SYMBOL === 'BEP20' ?["BNB", "USDT"] : res.SYMBOL === 'ERC20' ?["ETH",

"USDT"

] : res.SYMBOL === 'TRC20' ?["TRX", "USDT"] : []

// TOKENLIST: ["USDT"]

}

if (walletInfo. NAME = = = 'My wallet is')

walletInfo.NAME = this.getNewWalletName(walletInfo)

if (!uni.getStorageSync("appPwd"))

uni.setStorageSync("appPwd", that.pwd)

that.saveWallet(walletInfo)

that.setSelectChain(walletInfo.SYMBOL)

that.setSelectWallet(walletInfo)

uni.navigateTo({

url: '/pages/create/importLoading'

})

} else {

uni.showToast({

Title: 'Import wallet failed, reason:' + res.data.MESSAGE,

duration: 3000,

icon: "none"

})

}

}

}

}

</script>

<style lang="scss">

page {

height: 100%;

background-color: #F2F3F7;

}

</style>

<style lang="scss" scoped>

.status-bar {

width: 100%;

background: #F2F3F7;

}

.nav {

padding: 0 26rpx;

margin-top: 20rpx;

.nav-left {

text-align: center;

position: relative;

display: flex;

align-items: center;

.icon {

position: absolute;

left: 0;

}

text {

font-size: 36rpx;

font-weight: bold;

color: #000000;

margin: 0 auto;

}

.icon-scan {

position: absolute;

right: 0;

width: 42rpx;

height: 42rpx;

}

}

}

.type {

height: 100rpx;

line-height: 100rpx;

text-align: center;

background: #EAEBF0;

border-radius: 20rpx;

margin: 0 30rpx;

margin-top: 42rpx;

padding: 5rpx;

display: flex;

align-items: center;

.type-item {

width: 50%;

height: 100rpx;

border-radius: 20rpx;

font-size: 30rpx;

font-weight: bold;

color: #000000;

}

.type-item-active {

background: #ffffff;

}

}

.input-container {

position: relative;

.type-input {

margin: 40rpx 30rpx 0 30rpx;

width: auto;

height: 180rpx;

background: #FFFFFF;

border: 1px solid #EBEBEB;

border-radius: 20rpx;

padding: 60rpx 30rpx;

font-size: 26rpx;

}

text {

position: absolute;

right: 60rpx;

bottom: 30rpx;

font-size: 26rpx;

color: #6C2AEE;

}

}

.name {

margin: 40rpx 0 30rpx 30rpx;

font-size: 30rpx;

font-weight: bold;

color: #000000;

}

.input-name {

margin: 0 30rpx;

padding: 0 20rpx;

height: 100rpx;

background: #FFFFFF;

border-radius: 20rpx;

font-size: 26rpx;

}

.senior {

margin: 30rpx 30rpx 0 30rpx;

display: flex;

align-items: center;

.senior-name {

margin-left: auto;

margin-right: 12rpx;

font-size: 26rpx;

color: #6C2AEE;

}

}

.bottom {

width: -webkit-fill-available;

position: fixed;

-webkit-transform: translateZ(0);

transform: translateZ(0);

bottom: env(safe-area-inset-bottom);

bottom: constant(safe-area-inset-bottom);

padding: 0 30rpx;

margin-bottom: 80rpx;

overflow: hidden;

.btn-confirm {

background: #6C2AEE;

border-radius: 20rpx;

font-size: 36rpx;

font-weight: bold;

color: #FFFFFF;

}

}

</style>

<template>

<view class="content">

<image class="logo" src="/static///logo\_loading.gif"></image>

<view class= "loading-title"> Importing backbone data

<view class="dot"></view>

</view>

<view class="loading-msg">

Backup of your private key and mnemonic words

</view>

<view class="loading-bottom">

<image class="bottom-logo" src="../../static/img\_logo\_small.png"></image>

<view class="name">

PYY wallet

</view>

</view>

</view>

</template>

<script>

export default {

data() {

return {};

},

onLoad() {

setTimeout(function() {

uni.switchTab({

url: "/pages/wallet/index"

})

}, 1500);

},

methods: {}

}

</script>

<style lang="scss" scoped>

.content {

display: flex;

flex-direction: column;

align-items: center;

.logo {

margin-top: 480rpx;

width: 120rpx;

height: 150rpx;

}

.loading-title {

margin-top: 60rpx;

font-size: 36rpx;

font-weight: 500;

color: #000000;

.dot {

display: inline-block;

height: 1.2em;

line-height: 1;

text-align: left;

vertical-align: -.25em;

overflow: hidden;

}

.dot::before {

display: block;

content: '...\A..\A.';

white-space: pre-wrap;

/\* Can also be a white-space: pre \*/

animation: dot 1s infinite step-start both;

}

@keyframes dot {

33% {

transform: translateY(-2em);

}

66% {

transform: translateY(-1em);

}

}

}

.loading-msg {

margin-top: 50rpx;

font-size: 24rpx;

font-weight: 500;

color: #7729FB;

}

.loading-bottom {

display: flex;

align-items: center;

position: fixed;

bottom: 120rpx;

.bottom-logo {

width: 60rpx;

height: 80rpx;

}

.name {

margin-left: 18rpx;

font-size: 36rpx;

font-weight: bolder;

color: #1B162F;

}

}

}

</style>

<template>

<view class="content">

<image class="logo" src="/static///logo\_loading.gif"></image>

<view class= "loading-title"> Importing backbone data

<view class="dot"></view>

</view>

<view class="loading-msg">

Backup of your private key and mnemonic words

</view>

<view class="loading-bottom">

<image class="bottom-logo" src="../../static/img\_logo\_small.png"></image>

<view class="name">

PYY wallet

</view>

</view>

</view>

</template>

<script>

import {

createAddress

} from "@/api/wallet.js"

export default {

data() {

return {

pwd: '',

createType: 'BEP 20' / / Default Binance chain

};

},

onLoad(option) {

var that = this

this.pwd = option.pwd

console.log(option);

this.createType = option.createType ?option.createType : this.createType

// The Binance chain wallet is created by default

createAddress(this.createType).then(res => {

if (res.STATUS === 200) {

// Cache the wallet password

if (!uni.getStorageSync("appPwd"))

uni.setStorageSync("appPwd", that.pwd)

// Cache the wallet information

let walletInfo = {

HEADLOGO: require("@/static/head/head\_photo\_" + Math.floor(Math.random() \* 10 + 1) +

".jpeg"),

PRIKEY: res. PRIKEY, / / Wallet key

MNEMONIC: res. MNEMONIC, / / mnemonic words

SYMBOL: res. SYMBOL, / / Main chain

ADDRESS: res. ADDRESS, / / wallet address

NAME: 'My wallet', / / wallet name

BACKUPS: false, / / backup status

TOKENLIST: res.SYMBOL === 'BEP20' ?["BNB", "USDT"] : res.SYMBOL === 'ERC20' ?["ETH",

"USDT"

] : res.SYMBOL === 'TRC20' ?["TRX", "USDT"] : []

// TOKENLIST: ["USDT"]

}

if (walletInfo. NAME = = = 'My Wallet')

walletInfo.NAME = this.getNewWalletName(walletInfo)

that.saveWallet(walletInfo)

that.setSelectChain(walletInfo.SYMBOL)

that.setSelectWallet(walletInfo)

setTimeout(function() {

uni.switchTab({

url: "/pages/wallet/index"

})

}, 2000);

} else {

uni.showToast({

Title: 'wallet creation failed, reason:' + res.data.MESSAGE,

duration: 3000,

icon: "none"

})

setTimeout(function() {

uni.navigateTo({

url: "/pages/index/index"

})

}, 300);

}

}).catch(err => {

console.log(err);

})

},

methods: {}

}

</script>

<style lang="scss" scoped>

.content {

display: flex;

flex-direction: column;

align-items: center;

.logo {

margin-top: 480rpx;

width: 120rpx;

height: 150rpx;

}

.loading-title {

margin-top: 60rpx;

font-size: 36rpx;

font-weight: 500;

color: #000000;

.dot {

display: inline-block;

height: 1.2em;

line-height: 1;

text-align: left;

vertical-align: -.25em;

overflow: hidden;

}

.dot::before {

display: block;

content: '...\A..\A.';

white-space: pre-wrap;

/\* Can also be a white-space: pre \*/

animation: dot 1s infinite step-start both;

}

@keyframes dot {

33% {

transform: translateY(-2em);

}

66% {

transform: translateY(-1em);

}

}

}

.loading-msg {

margin-top: 50rpx;

font-size: 24rpx;

font-weight: 500;

color: #7729FB;

}

.loading-bottom {

display: flex;

align-items: center;

position: fixed;

bottom: 120rpx;

.bottom-logo {

width: 60rpx;

height: 80rpx;

}

.name {

margin-left: 18rpx;

font-size: 36rpx;

font-weight: bolder;

color: #1B162F;

}

}

}

</style>

<template>

<view>

<!--Status bar, -->

<view class="status\_bar" :style="{height:topHeight+'px'}"></view>

<!--Navigation bar, -->

<uni-nav-bar left-icon="left" @clickLeft="onClickGoBack" :border="false">

<view class="nav-title">

<view class="title-content">

<Text> Create a wallet </ text>

<view class="title-bottom">

<view class="bottom-split" :class="{'split-active':step===1 || step===2}"></view>

<view class="bottom-split" :class="{'split-active':step===2}"></view>

</view>

</view>

</view>

</uni-nav-bar>

<!--Page body-->

<view class="content">

<view class= "body-title"> Verify Password </ view>

<view class="body-msg">

When you forget the transaction password, you can uninstall and reload into your mnemonic

Words, in order to redesign the transaction password

</view>

<view class="body-msg-tip">

Set the 6-digit plain digital password

<uni-icons v-show="!isShowPwd" type="eye-slash" size="20" @click="onClickEye"></uni-icons>

<uni-icons v-show="isShowPwd" type="eye" size="20" @click="onClickEye"></uni-icons>

</view>

<input class="cinput" adjust-position="false" auto-blur="true" @blur="blur" @input="codenum" :focus="focus"

value="pwd" v-model="pwd" type="tel" maxlength="6" :cursor="pwd.length" />

<view class="code-input">

<view v-for="(item,index) in 6" :key="index" @click="codefocus(index)"

:style='(index == pwd.length?"border: 2rpx solid #7729FB;":"color: " + codeColor + ";" +"border: 2rpx solid #EBEBEB")'>

{{showPassword(index)}}

</view>

</view>

<view v-show="!isPwdVerify" class="body-msg-tip" style="margin-top: 20rpx;color: red">

Two input password is inconsistent, please re-enter ~

</view>

</view>

</view>

</template>

<script>

export default {

data() {

return {

title: 'Hello',

topHeight: 0,

step: 1,

Fus: true, / / input focus for keyboard hiding and reocation

codeColor: "# 313131", / / Color of the custom cursor

Pwd: '', / / This is the password entered by the user,

oldPwd: '', / / Password entered for the first time

isShowPwd: False, / / whether the password is displayed

isPwdVerify: true, / / is the password entered twice consistent \

type: 'create'

}

},

onLoad(option) {

let topHeight = uni.getSystemInfoSync().statusBarHeight

this.topHeight = topHeight

this.oldPwd = option.pwd

this.type = option.type

},

methods: {

onClickGoBack() {

uni.navigateBack();

},

onClickEye() {

this.isShowPwd = !this.isShowPwd

},

showPassword(index) {

if (this.pwd[index] === undefined)

return ''

return this.isShowPwd ?this.pwd[index] : "●"

},

// enter password

codenum: function(event) {

console.log(event);

var that = this

var pwd = event.target.value.replace(/\D/g, '')

that.pwd = pwd

console.log(pwd);

// Check whether the password entered twice is consistent

if (pwd.length == 6) {

if (this.oldPwd !== this.pwd) {

this.isPwdVerify = false

setTimeout(function() {

that.pwd = ''

that.isPwdVerify = true

}, 1000);

return

}

if (this.type === 'create') {

uni.navigateTo({

url: '../create/loading?pwd=' + this.pwd

})

} else if (this.type === 'import') {

uni.navigateTo({

url: '../create/import?pwd=' + this.pwd

})

}

this.pwd = ''

}

},

// The keyboard is hidden after the setting loses focus

blur: function() {

var that = this

that.focus = false

},

// Click on the custom cursor to display the keyboard

codefocus: function(e) {

var that = this

this.focus = true

}

}

}

</script>

<style lang="scss" scoped>

.status\_bar {

width: 100%;

}

.nav-title {

width: 100%;

display: flex;

align-items: center;

justify-content: center;

.title-content {

display: flex;

flex-direction: column;

align-items: center;

text {

text-align: center;

font-size: 36rpx;

font-weight: bold;

color: #000000;

}

.title-bottom {

display: flex;

.bottom-split {

margin: 2px;

width: 60rpx;

height: 6rpx;

background: #E6E6E6;

border-radius: 2rpx;

}

.split-active {

background: #7729FB;

}

}

}

}

.content {

display: flex;

flex-direction: column;

padding-left: 80rpx;

.body-title {

margin-top: 90rpx;

font-size: 48rpx;

font-family: Source Han Sans CN;

font-weight: bold;

color: #000000;

}

.body-msg {

margin-top: 30rpx;

margin-right: 60rpx;

font-size: 28rpx;

font-weight: 400;

color: #000000;

}

.body-msg-tip {

margin-top: 100rpx;

font-size: 28rpx;

font-weight: 400;

color: #000000;

display: flex;

align-items: center;

.uni-icons {

margin-left: 20rpx;

}

}

}

.code-input {

display: flex;

}

.code-input>view {

margin-top: 30rpx;

margin-right: 16rpx;

width: 86rpx;

height: 86rpx;

line-height: 86rpx;

font-size: 48rpx;

font-weight: bold;

color: #313131;

text-align: center;

border-radius: 10rpx;

background: #F8F8F8;

}

.code-input>view:nth-child(1) {

margin-left: 0rpx;

}

.cinput {

position: fixed;

left: -100rpx;

width: 50rpx;

height: 50rpx;

}

</style>