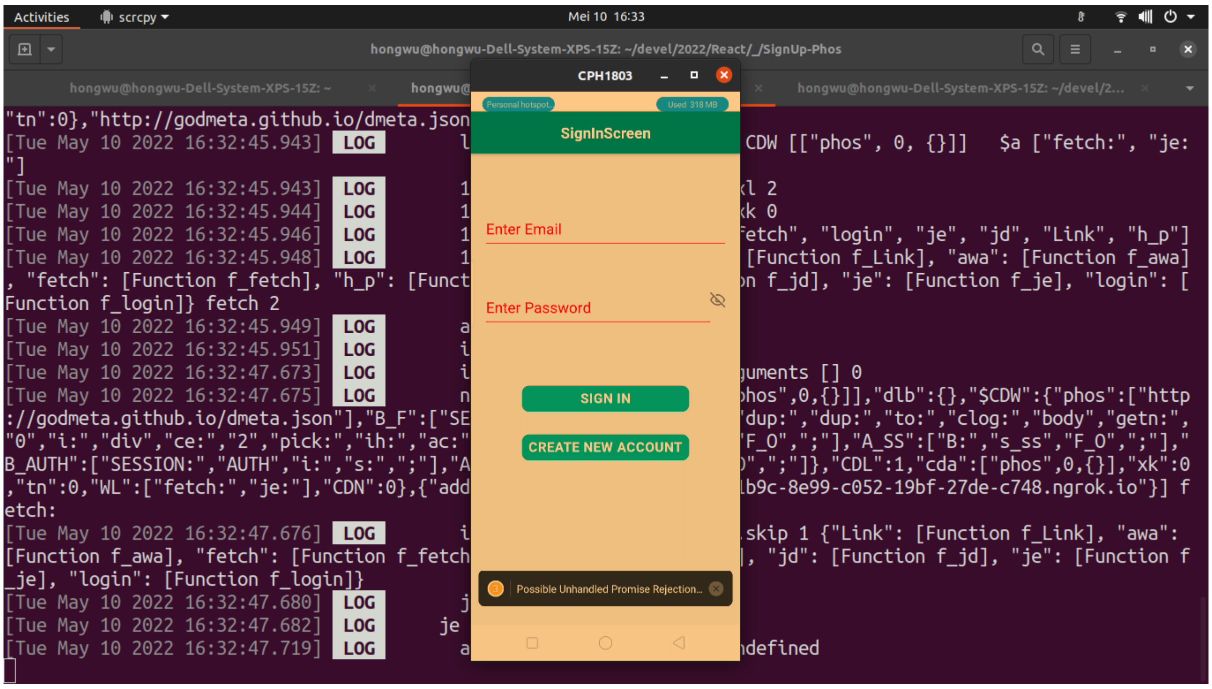
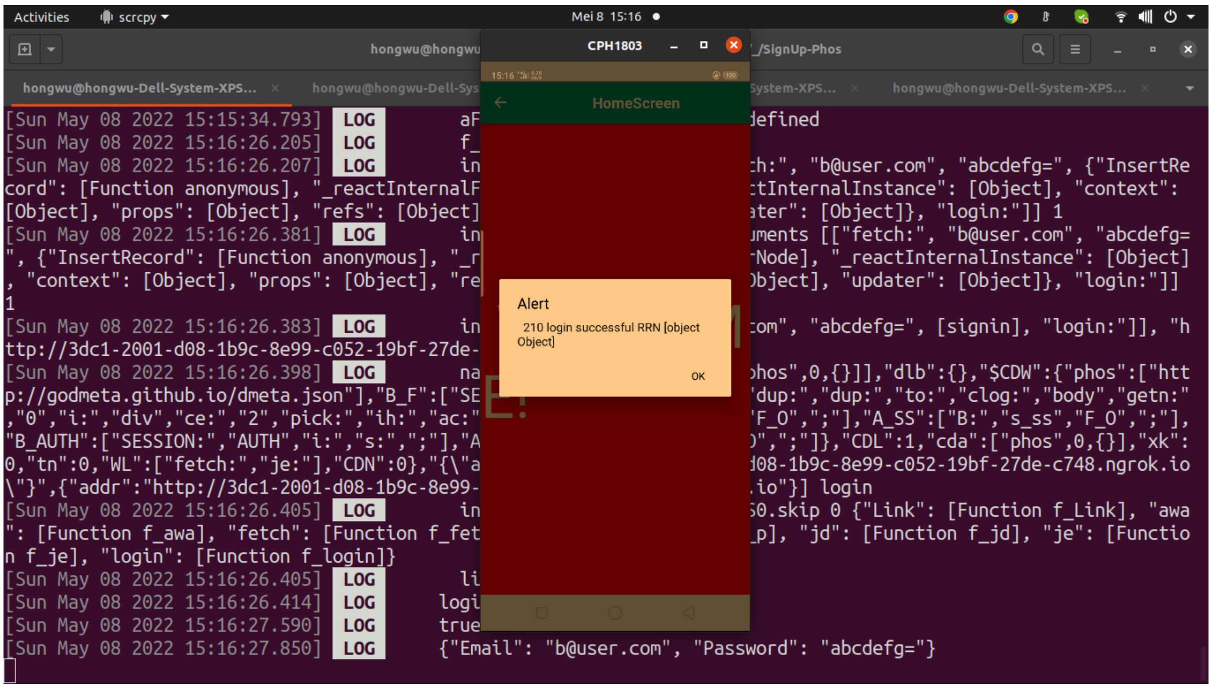
DSA001 DMeta SuperApp - Login

Figure 1: Initial login screen on DMeta SuperApp
running on Android device, viewed using scrcpy
and React Native running in the background terminal.


Activities
scrcpy
Mei 10 16:33
hongwu@hongwu-Dell-System-XPS-15Z: ~/devel/2022/React/_/SignUp-Phos
hongwu@hongwu-Dell-System-XPS-15Z: ~
hongwu@hongwu-Dell-System-XPS-15Z: ~/devel/2...
CPH1803
Personal hotspot...
Used 318 MB
SignInScreen
Enter Email
Enter Password
SIGN IN
CREATE NEW ACCOUNT
Possible Unhandled Promise Rejection...

```
"tn":0},"http://godmeta.github.io/dmeta.json
[Tue May 10 2022 16:32:45.943]  LOG        l        CDW [["phos", 0, {}]]   $a ["fetch:", "je:
"]
[Tue May 10 2022 16:32:45.943]  LOG        1        kl 2
[Tue May 10 2022 16:32:45.944]  LOG        1        kk 0
[Tue May 10 2022 16:32:45.946]  LOG        1        etch", "login", "je", "jd", "Link", "h_p"]
[Tue May 10 2022 16:32:45.948]  LOG        1        [Function f_Link], "awa": [Function f_awa]
, "fetch": [Function f_fetch], "h_p": [Funct        on f_jd], "je": [Function f_je], "login": [
Function f_login]} fetch 2
[Tue May 10 2022 16:32:45.949]  LOG        a
[Tue May 10 2022 16:32:45.951]  LOG        i
[Tue May 10 2022 16:32:47.673]  LOG        i        guments [] 0
[Tue May 10 2022 16:32:47.675]  LOG        n        hos",0,{}]],"dlb":{},"$CDW":{"phos":["http
://godmeta.github.io/dmeta.json"],"B_F":["SE        'dup:","dup:","to:","clog:","body","getn:",
"0","i:","div","ce:","2","pick:","ih:","ac:"        'F_0",";"],"A_SS":["B:","s_ss","F_0",";"],"
B_AUTH":["SESSION:","AUTH","i:","s:",";"],"A        )",";"]},"CDL":1,"cda":["phos",0,{}],"xk":0
,"tn":0,"WL":["fetch:","je:"],"CDN":0},{"add        1b9c-8e99-c052-19bf-27de-c748.ngrok.io"}] f
etch:
[Tue May 10 2022 16:32:47.676]  LOG        i        skip 1 {"Link": [Function f_Link], "awa":
[Function f_awa], "fetch": [Function f_fetch        ], "jd": [Function f_jd], "je": [Function f
_je], "login": [Function f_login]}
[Tue May 10 2022 16:32:47.680]  LOG        j
[Tue May 10 2022 16:32:47.682]  LOG       je
[Tue May 10 2022 16:32:47.719]  LOG        a        ndefined
```

Figure 2: Login successfully.

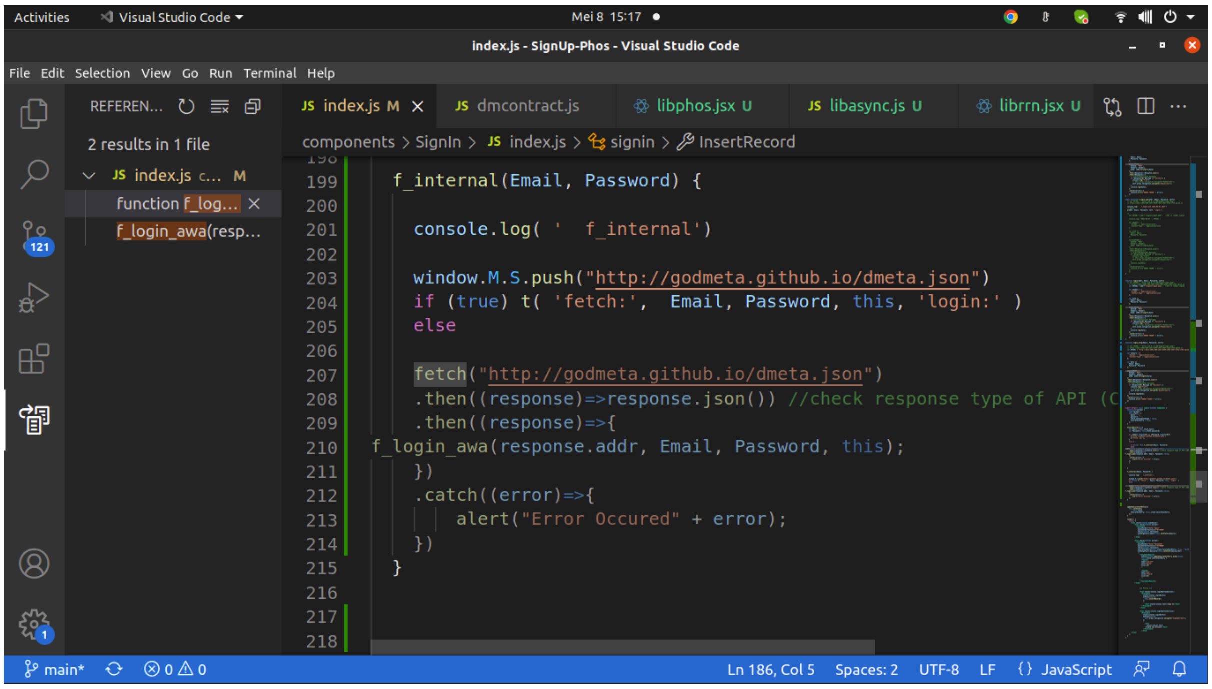


Alert

210 login successful RRN [object Object]

```
[Sun May 08 2022 15:15:34.793]  LOG        aF       defined
[Sun May 08 2022 15:16:26.205]  LOG        f_
[Sun May 08 2022 15:16:26.207]  LOG        in       h:", "b@user.com", "abcdefg=", {"InsertRe
cord": [Function anonymous], "_reactInternalF       tInternalInstance": [Object], "context":
[Object], "props": [Object], "refs": [Object]       ater": [Object]}, "login:"]] 1
[Sun May 08 2022 15:16:26.381]  LOG        in       ments [["fetch:", "b@user.com", "abcdefg=
", {"InsertRecord": [Function anonymous], "_r       Node], "_reactInternalInstance": [Object]
, "context": [Object], "props": [Object], "re       bject], "updater": [Object]}, "login:"]]
1
[Sun May 08 2022 15:16:26.383]  LOG        in       om", "abcdefg=", [signin], "login:"]], "h
ttp://3dc1-2001-d08-1b9c-8e99-c052-19bf-27de-
[Sun May 08 2022 15:16:26.398]  LOG        na       hos",0,{}]],"dlb":{},"$CDW":{"phos":["htt
p://godmeta.github.io/dmeta.json"],"B_F":["SE       'dup:","dup:","to:","clog:","body","getn:"
,"0","i:","div","ce:","2","pick:","ih:","ac:"       'F_0",";"],"A_SS":["B:","s_ss","F_0",";"],
"B_AUTH":["SESSION:","AUTH","i:","s:",";"],"A       )",";"]},"CDL":1,"cda":["phos",0,{}],"xk":
0,"tn":0,"WL":["fetch:","je:"],"CDN":0},"{\"a       d08-1b9c-8e99-c052-19bf-27de-c748.ngrok.io
\"}",{"addr":"http://3dc1-2001-d08-1b9c-8e99-       .io"}] login
[Sun May 08 2022 15:16:26.405]  LOG        in       50.skip 0 {"Link": [Function f_Link], "awa
": [Function f_awa], "fetch": [Function f_fet       p], "jd": [Function f_jd], "je": [Functio
n f_je], "login": [Function f_login]}
[Sun May 08 2022 15:16:26.405]  LOG        li
[Sun May 08 2022 15:16:26.414]  LOG      logi
[Sun May 08 2022 15:16:27.590]  LOG      true
[Sun May 08 2022 15:16:27.850]  LOG      {"Email": "b@user.com", "Password": "abcdefg="}
```

Figure 3: Proscription Reverse React Notation one liner (line 204)
vs. original React code (lines 207 to 214)


Activities
Visual Studio Code
Mei 8 15:17
index.js - SignUp-Phos - Visual Studio Code
File Edit Selection View Go Run Terminal Help
REFEREN...
2 results in 1 file
index.js c... M
function f_log...
f_login_awa(resp...
index.js M
dmcontract.js
libphos.jsx U
libasync.js U
librrn.jsx U
components > SignIn > index.js > signin > InsertRecord
main*
0 0
Ln 186, Col 5 Spaces: 2 UTF-8 LF {} JavaScript

```
199      f_internal(Email, Password) {
200
201        console.log( '  f_internal')
202
203        window.M.S.push("http://godmeta.github.io/dmeta.json")
204        if (true) t( 'fetch:',  Email, Password, this, 'login:' )
205        else
206
207        fetch("http://godmeta.github.io/dmeta.json")
208        .then((response)=>response.json()) //check response type of API (C
209        .then((response)=>{
210    f_login_awa(response.addr, Email, Password, this);
211        })
212        .catch((error)=>{
213          alert("Error Occured" + error);
214        })
215      }
216
217
218
```

DMeta SuperApp: Login

DMeta SuperApp is the name of the mobile plus web architecture and application created based on DMeta server and Phoscript, a metaprogramming script derived from FORTH programming language, that aims to achieve the following:

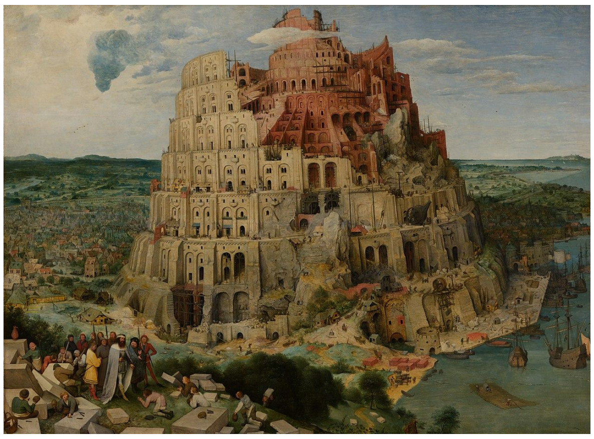
- Simplifying mobile and web apps programming, from lengthy blocky codes that typically span over 10 lines to FORTH or Phoscript one liners.
- Providing a platform where programmers can collaborate and CO-OWN source code, avoiding the need of rewriting an app from scratch, so that they may build a SuperApp together literally ONE LINE AT A TIME using Phoscript.
- Programmers use hash to identify code at function level, gaining greater control while negotiating with investors.
- Hash of Phoscript Code (HPC) allows portability of algorithms across multiple programming languages.
- Reference: DMW101 DMeta SuperApp
    - https://godmeta.github.io/doc/DMW101_DMeta_SuperApp.pdf

In short, DMeta SuperApp could be the biggest breakthrough in free software or open source ecosystems since GCC and Linux, allowing programmers to own and control the use of source code DOWN TO ONE LINE OF CODE.

In this tutorial, we show (figure 3) how React source code (lines 207 to 214) can be simplified to one single line of Phoscript Reverse React Notation (line 204).

The original source repository can be found in:

- DAR001 DMeta Mobile App using Reverse React Notation – Tunnel Redirection
- https://docs.google.com/document/d/13OraMz9NLU4Uf9-Kso-OHbAwbzdJq3S-lse0r98N1u4/edit?usp=drivesdk
- https://godmeta.github.io/doc/DAR001_DMeta_App_RRN_Tunnel_Redirection.pdf



The Tower of Babel and the Language of Light – Phoscript

If your grandfather is still alive, would you ask if his grandfather understood what the atomic bomb, television and mobile phones were, before they were invented?

In fact, legends have it that they were recorded in ancient sacred texts in the Orient.

If we tell you that a universal language that is capable of uniting all humankind has been invented in the 1960s, would you believe it?

This time, the legend of all humankind united by one language was recorded in ancient sacred texts not in the Orient, but the near East, yet, written in the reverse order.

With this language of Light, or Phoscript, derived from FORTH and reverse polish notation, humankind may reverse the effect of the Tower of Babel.

And as they said, the rest is history, only this time, YOU will be writing it ….

https://en.m.wikipedia.org/wiki/Tower_of_Babel

- The Tower of Babel (Hebrew: מִגְדַּל בָּבֶל, Migdal Bavel) narrative in Genesis 11:1–9 is an origin myth meant to explain why the world's peoples speak different languages.[1][2][3][4]

- According to the story, a united human race in the generations following the Great Flood, speaking a single language and migrating eastward, comes to the land of Shinar (שִׁנְעָר). There they agree to build a city and a tower tall enough to reach heaven. God, observing their city and tower, confounds their speech so that they can no longer understand each other, and scatters them around the world.

React / Javascript functions related to Phoscript Reverse React Notation
(full source code is available upon request)

```
f_internal(Email, Password) {
   window.M.S.push("http://godmeta.github.io/dmeta.json")
   if (true) t( 'fetch:',  Email, Password, this, 'login:' )
   else
      fetch("http://godmeta.github.io/dmeta.json")
   .then((response)=>response.json()) //check response type of API
(CHECK OUTPUT OF DATA IS IN JSON)
   .then((response)=>{
      f_login_awa(response.addr, Email, Password, this);
   })
   .catch((error)=>{
      alert("Error Occured" + error);
   })
 }
```

```
   t( 'fetch:',  Email, Password, this, 'login:' )
```

```
async function t() { // call async functions with then
   var $l = strlen(arguments[0]);
   var $fn = substr(arguments[0], 0, $l - 1);
   if (in_array($fn, array_keys(M.AF))) {
       M.AF[$fn](arguments)
   }
}
```

```
async function f_fetch() {
   var S0 = window.M.S[0]
   var M = window.M
   await fetch(window.M.S.pop())
   .then((response)=>response.json()) //check response type of API
(CHECK OUTPUT OF DATA IS IN JSON)
   .then((response)=>{
       var S = window.M.S;
       window.M.S.push(response)
       if (arguments.length==0) n()
       else na(arguments, response.addr)
       return response;
   })
   .catch((error)=>{ alert("f_fetch Error Occured: " + error);})
}
```

```
// na next word in then function with arguments
function na() {
   var S0 = window.M.S[0]
   var M = window.M
   var S = window.M.S;

   var A=arguments;
   var c=arguments[0][0].length;
   var l=arguments[0][0][c-1].length;
   var $fn=arguments[0][0][c-1].substring(0,l-1);

   S.push(A[A.length-1])
   S.push(A[0][0][1])
   S.push(A[0][0][2])
   S.push(A[0][0][3])

   if (in_array($fn, array_keys(M.CF))) {
     M.CF[$fn]();
   }
   return S[S.length-1];
}
```

```
async function f_login(){
     var S = window.M.S;
     var self=S.pop()
     var Password=S.pop()
     var Email=S.pop()
     var addr=S.pop()
     var APIURL = addr+"/SignIn/login.php";   //API to render signup
     var headers = {
       'Accept' : 'application/json',
       'Content-Type' : 'application/json'
     };
     var Data ={
       Email: Email,
       Password: Password
     };
     // MUST have await!!
     await fetch(APIURL,{
       method: 'POST',
       headers: headers,
       body: JSON.stringify(Data)
     })
     .then((Response)=>Response.json())
     .then((Response)=>{
       if (Response[0].Message == "Success") {
         self.props.navigation.navigate("HomeScreen");
       }
       console.log(Data);
     })
     .catch((error)=>{
       console.error("ERROR FOUND" + error);
     })
 }
```