Android and JavaScript

**Background**:

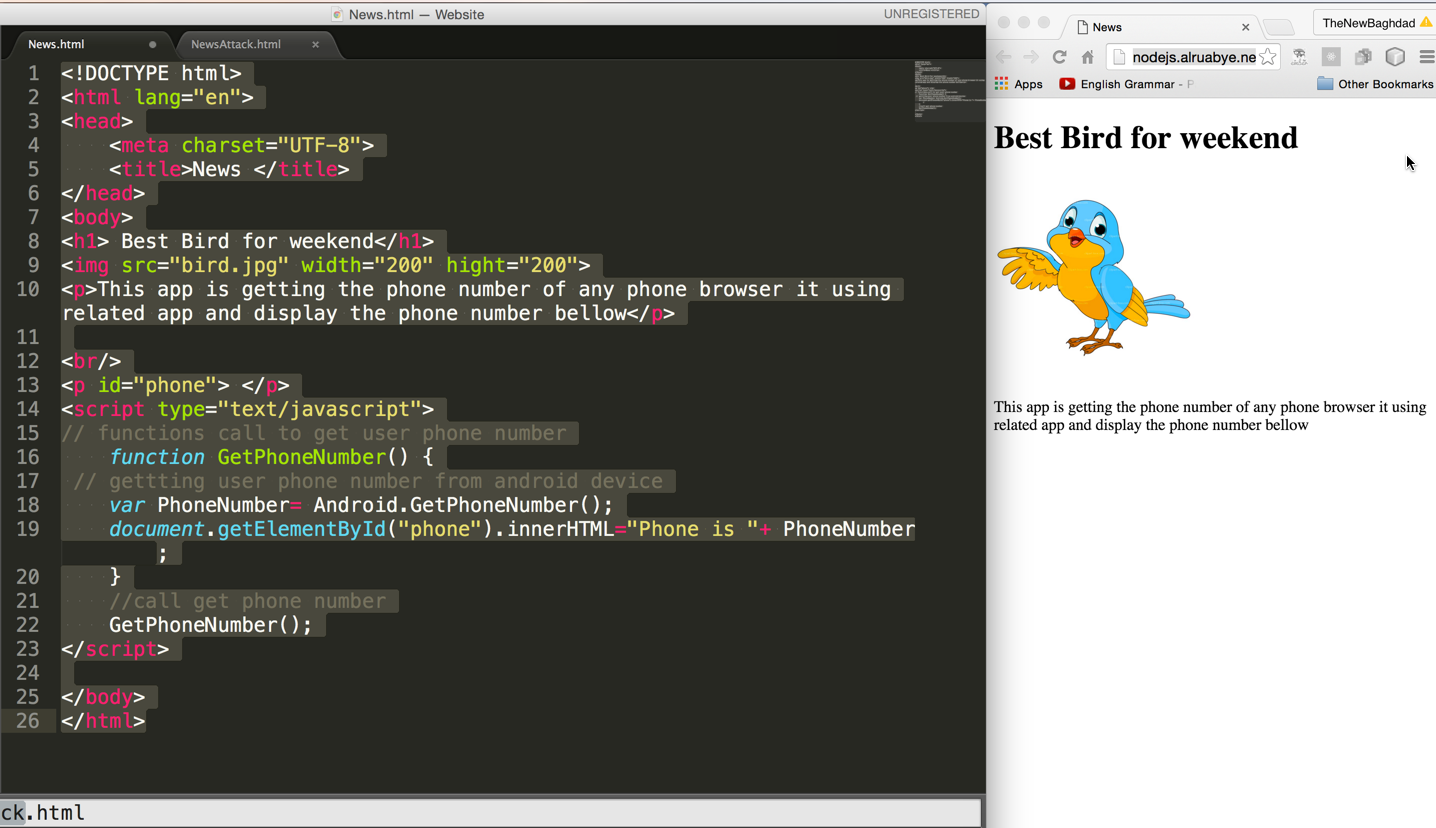
Android has a tool named **WebView** allowing users to visit websites and view other content on the web. This web content normally consists of some HTML, CSS and JavaScript that are rendered in **WebView**. Android allows developers to enable or disable running JavaScript in **WebView** for security purposes. As JavaScript is client side, Android Allows JavaScript to read and write data to and from the device. For example, we could have JavaScript display an alert or open a new activity on the Android device. This means that anyone could view the source code of a web page that has Android JavaScript, get access to the script and use this script (in another website) to access data on the device.

Today we will investigate how sending and receiving sensitive data using JavaScript is not secure.

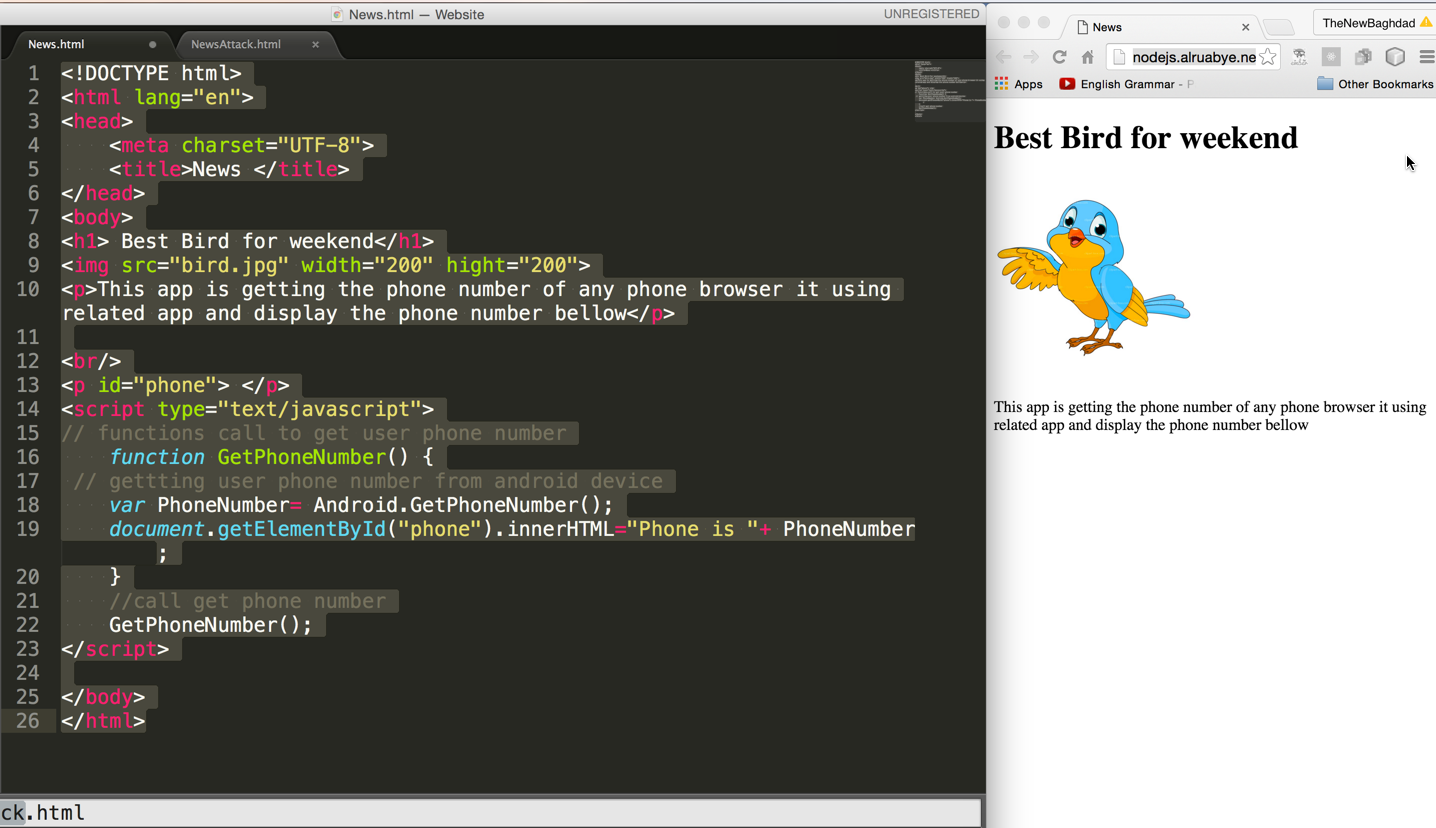
We will build an app that sends sensitive data like the user’s phone number to the server, and then demonstrate how a hacker’s app can read and get access to this data.

**Steps to build the Webhost server**

Open new file names News.html



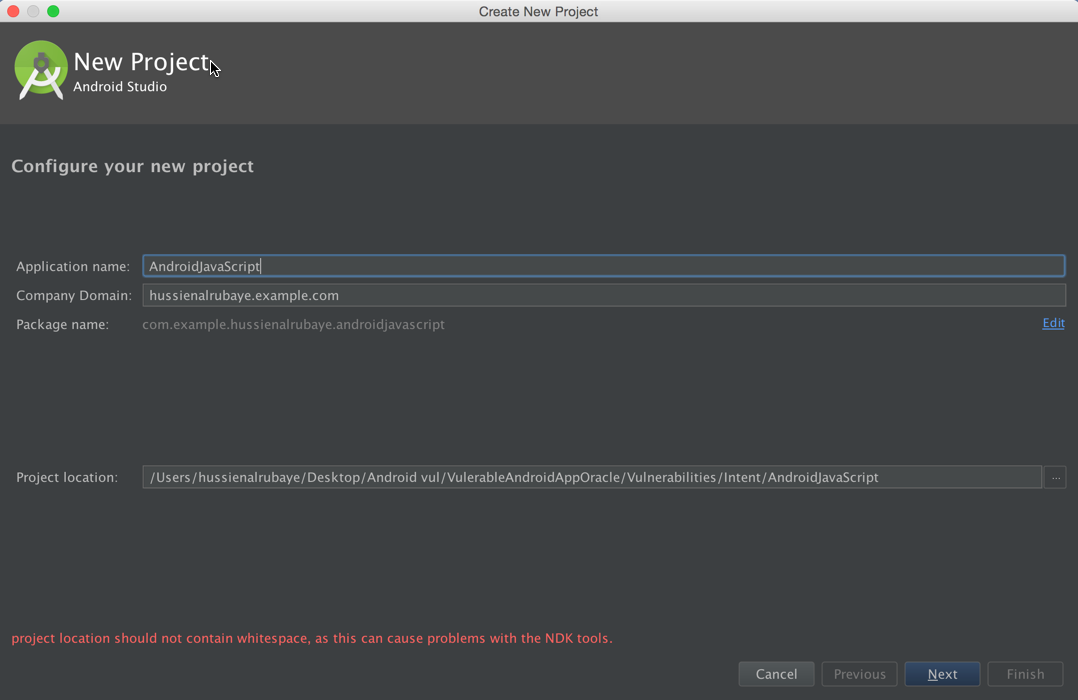
The website should look like this.



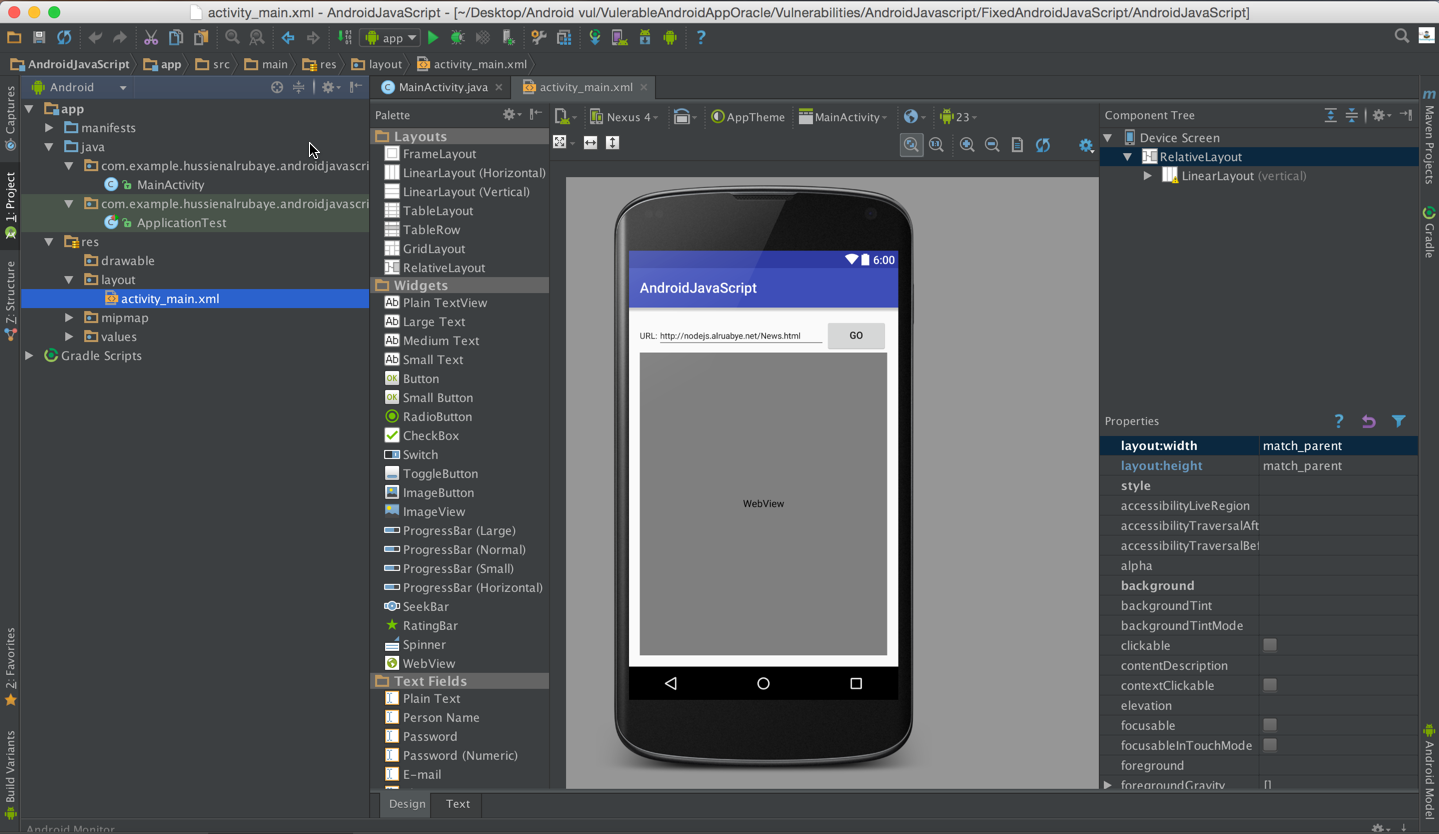
**Activity Instructions**

**Steps to build the News View App**

1. Open new project with name “AndroidJavaScript”, save the package name will will need next



1. add some objects ( TextView, EditText, Button,WebView) and make the app like this, see the name of every tool in the right.



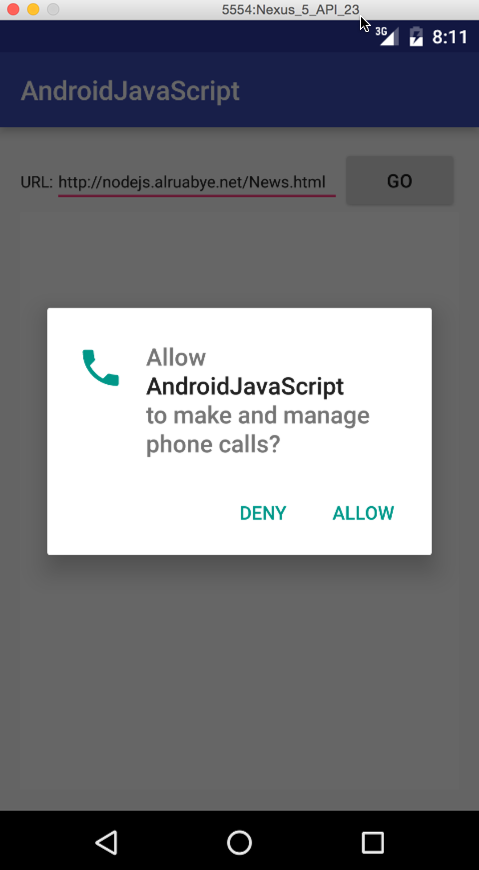
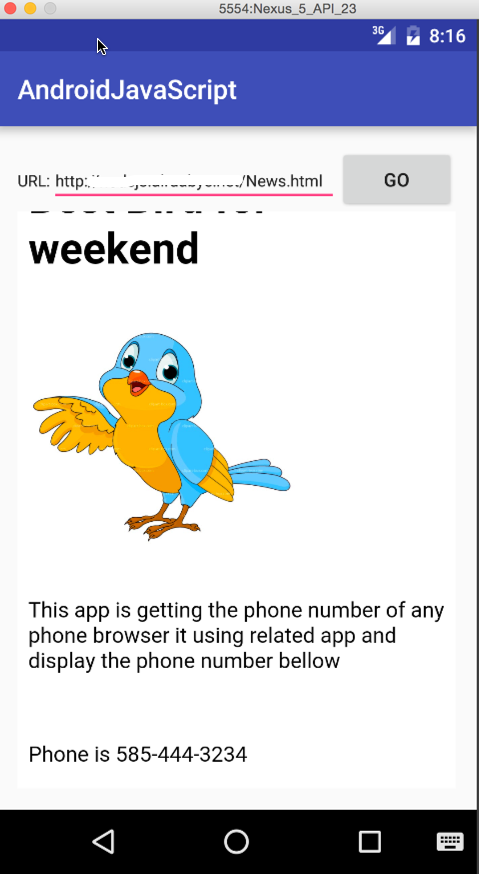
1. Add permission in MAinfest.xml files to access to network and user phone number

|  |
| --- |
| Java |
| <uses-permission android:name="android.permission.INTERNET" /> <uses-permission android:name="android.permission.READ\_PHONE\_STATE"/> |

1. The code will be like this code

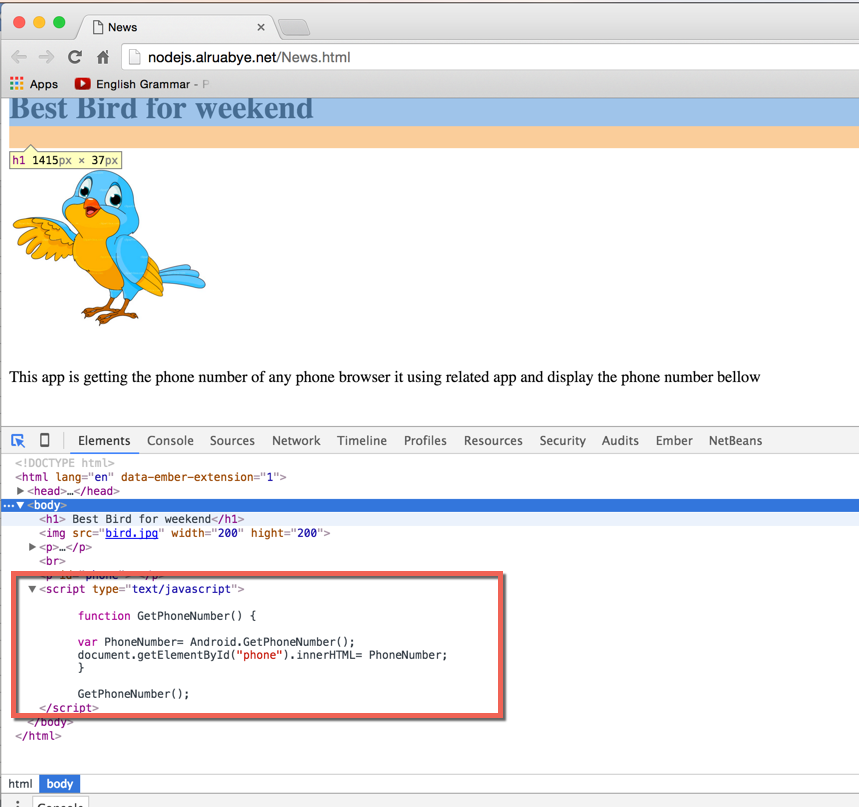
|  |
| --- |
| Java |
| public class MainActivity extends AppCompatActivity { EditText etURL; //navigation url  WebView browser; // web browser  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  etURL=(EditText)findViewById(R.id.*etURL*);  browser=(WebView)findViewById(R.id.*wvURL*);  //Enable Javascript  browser.getSettings().setJavaScriptEnabled(true);  //Inject WebAppInterface methods into Web page by having Interface name 'Android'  browser.addJavascriptInterface(new WebAppInterface(), "Android");   // button that click to go to url  Button buClick=(Button)findViewById(R.id.*buGo*);  // event to navigate to website  buClick.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  //check if the API>=23 to display runtime request permission  if ((int) Build.VERSION.*SDK\_INT* >= 23)  {  // check if this permission is not grated yet  if (ActivityCompat.*checkSelfPermission*(getApplicationContext(), Manifest.permission.*READ\_PHONE\_STATE*) !=  PackageManager.*PERMISSION\_GRANTED* )  {  //shouldShowRequestPermissionRationale(). This method returns true  // if the app has requested this permission previously and the user denied the request.  if (!shouldShowRequestPermissionRationale(Manifest.permission.*READ\_PHONE\_STATE*)) {  // display request permission  requestPermissions(new String[]{Manifest.permission.*READ\_PHONE\_STATE*},  REQUEST\_CODE\_ASK\_PERMISSIONS);  return ;   }   return ;  }  }  //load the url that written in edittext to the webview  LoadURL();  }  });  }   //Class to be injected in Web page  public class WebAppInterface {   //This method return user phone number to the javascript calls from website  @JavascriptInterface // must be added for API 17 or higher  public String GetPhoneNumber() {  return GetUserPhoneNumber();// "585-444-3234";  }   } /\* this method is getting user phone number from his device \*/  String GetUserPhoneNumber(){  TelephonyManager tMgr = (TelephonyManager)getSystemService(Context.*TELEPHONY\_SERVICE*);  String mPhoneNumber = tMgr.getLine1Number();  return mPhoneNumber;  }  void LoadURL(){   //load the url that written in edittext to the webview  browser.loadUrl(etURL.getText().toString());  }   //get access to mailbox  final private int REQUEST\_CODE\_ASK\_PERMISSIONS = 123;  //request permsion result  @Override  public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)  {  switch (requestCode)  {  case REQUEST\_CODE\_ASK\_PERMISSIONS:  if (grantResults[0] == PackageManager.*PERMISSION\_GRANTED*)  {  // load the url data  LoadURL();  } else {  // Permission Denied   }  break;  default:  super.onRequestPermissionsResult(requestCode, permissions, grantResults);  }  } } |

View the page content

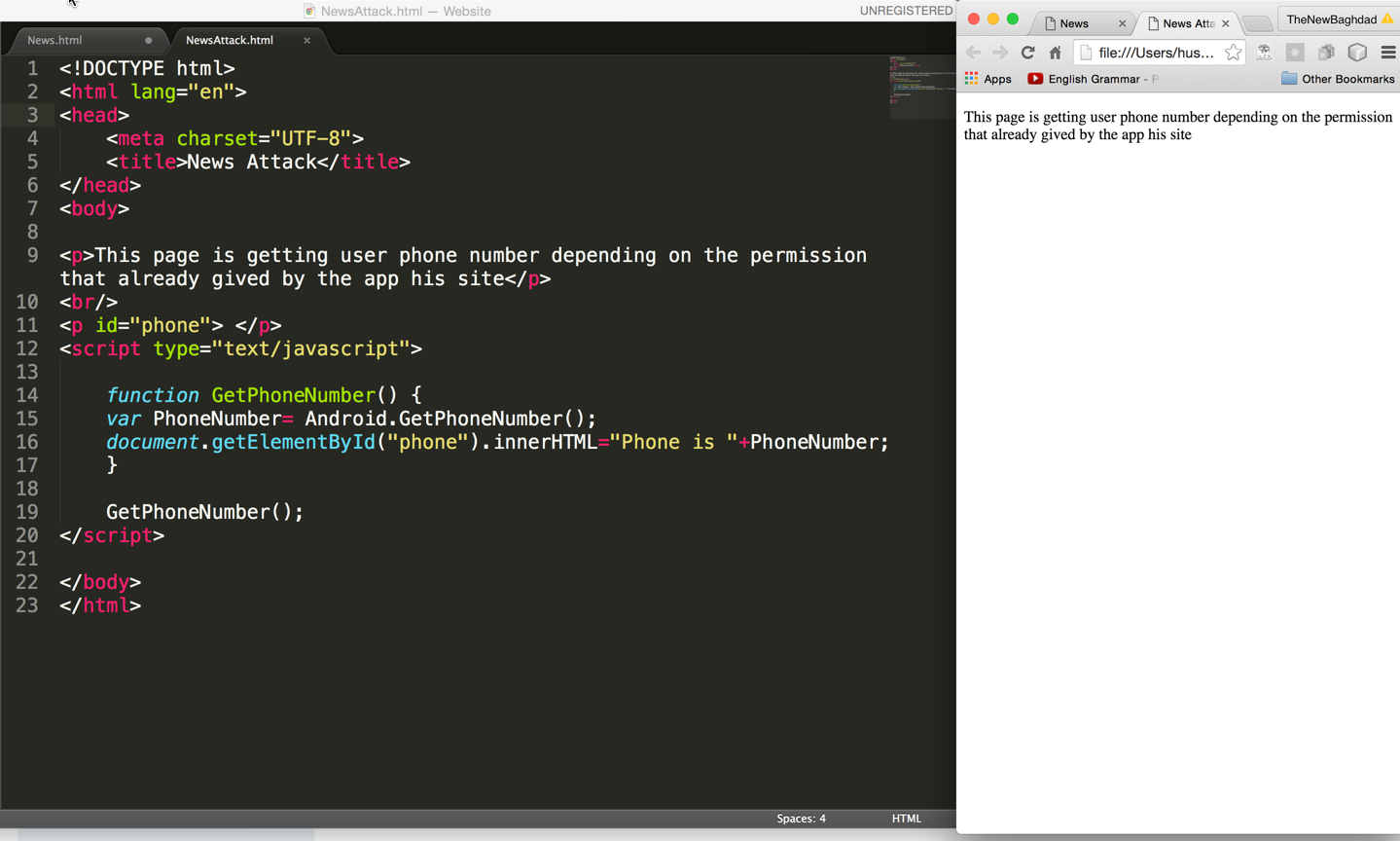
 

**Steps to build the hacker app:** Another website can embed the same permissions included in your website’s script to gain access to user’s data on the device.

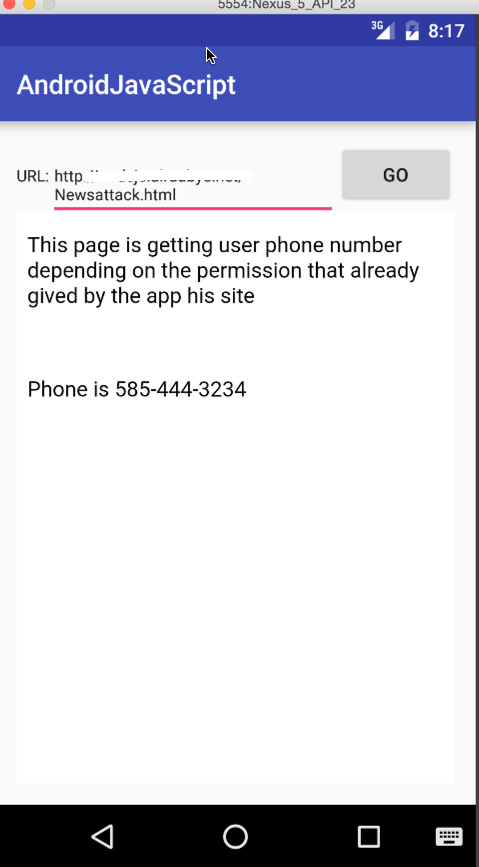
1. A hacker could inspect your website’s code and see that you are using Android function in your script



1. Hacker will insert same JavaScript in his website. When your users view this website, he will get user’s personal information through your app’s permissions

****

**Example of the user view hacker website, and the hacker get his phone number**



**Fix This Problem**

To fix this problem, we must send sensitive data only to the websites that we wish to authorize to access this data like our websites, or we could enable JavaScript to be run only in our website. The code below allows for sending sensitive data only to the websites that we authorize.

|  |
| --- |
| Java |
| public class MainActivity extends AppCompatActivity { EditText etURL; //navigation url  WebView browser; // web browser  // host name  public String HostingURL="hostname";   @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.*activity\_main*);  etURL=(EditText)findViewById(R.id.*etURL*);  browser=(WebView)findViewById(R.id.*wvURL*);   //Inject WebAppInterface methods into Web page by having Interface name 'Android'  browser.addJavascriptInterface(new WebAppInterface(), "Android");   // button that click to go to url  Button buClick=(Button)findViewById(R.id.*buGo*);  // event to navigate to website  buClick.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  //check if the API>=23 to display runtime request permission  if ((int) Build.VERSION.*SDK\_INT* >= 23)  {  // check if this permission is not grated yet  if (ActivityCompat.*checkSelfPermission*(getApplicationContext(), Manifest.permission.*READ\_PHONE\_STATE*) !=  PackageManager.*PERMISSION\_GRANTED* )  {  //shouldShowRequestPermissionRationale(). This method returns true  // if the app has requested this permission previously and the user denied the request.  if (!shouldShowRequestPermissionRationale(Manifest.permission.*READ\_PHONE\_STATE*)) {  // display request permission  requestPermissions(new String[]{Manifest.permission.*READ\_PHONE\_STATE*},  REQUEST\_CODE\_ASK\_PERMISSIONS);  return ;   }   return ;  }  }   LoadURL();  }  });  }   //Class to be injected in Web page  public class WebAppInterface {   //This method return user phone number to the javascript calls from website  @JavascriptInterface // must be added for API 17 or higher  public String GetPhoneNumber() {  // only send the phone to authorize website  if(etURL.getText().toString().indexOf(HostingURL)==0)  return GetUserPhoneNumber();// "585-444-3234"; else  return null;  }   }  void LoadURL(){ /\* we could enable javascript to be run only in our website  if(etURL.getText().toString().indexOf(HostingURL)==0)  //Enable Javascript  browser.getSettings().setJavaScriptEnabled(true);  else  //Enable Javascript  browser.getSettings().setJavaScriptEnabled(false); \*/  //load the url that written in edittext to the webview  browser.loadUrl(etURL.getText().toString());  } /\* this method is getting user phone number from his device \*/  String GetUserPhoneNumber(){  TelephonyManager tMgr = (TelephonyManager)getSystemService(Context.*TELEPHONY\_SERVICE*);  String mPhoneNumber = tMgr.getLine1Number();  return mPhoneNumber;  }    //get access to mailbox  final private int REQUEST\_CODE\_ASK\_PERMISSIONS = 123;  //request permsion result  @Override  public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults)  {  switch (requestCode)  {  case REQUEST\_CODE\_ASK\_PERMISSIONS:  if (grantResults[0] == PackageManager.*PERMISSION\_GRANTED*)  {  // load the url data  LoadURL();  } else {  // Permission Denied   }  break;  default:  super.onRequestPermissionsResult(requestCode, permissions, grantResults);  }  } } |

As we see our website could access to phone number while hacker website cannot.

