**DOCUMENTAÇÃO**

SUMÁRIO

[1 ANDROID STUDIO 3](#_Toc423798956)

[2 EMULADOR 4](#_Toc423798957)

[2.1 INSTALANDO E CONFIGURANDO O EMULADOR GENYMOTION 4](#_Toc423798958)

[2.1.1 Instalando um device 7](#_Toc423798959)

[2.2 CONFIGURAR EMULADOR COM O GOOGLE PLAY STORE 9](#_Toc423798960)

[3 API DE MAPAS COM O ANDROID STUDIO 13](#_Toc423798961)

[3.1 Começando um novo projeto 13](#_Toc423798962)

[3.2 Acitivity 15](#_Toc423798963)

[3.3 Escolha de um Activity 17](#_Toc423798964)

[3.4 Iniciando o projeto 19](#_Toc423798965)

[3.5 Imports 20](#_Toc423798966)

[3.6 Variáveis 22](#_Toc423798967)

[3.7 Alterações na Interface 22](#_Toc423798968)

[3.7.1 API KEY 23](#_Toc423798969)

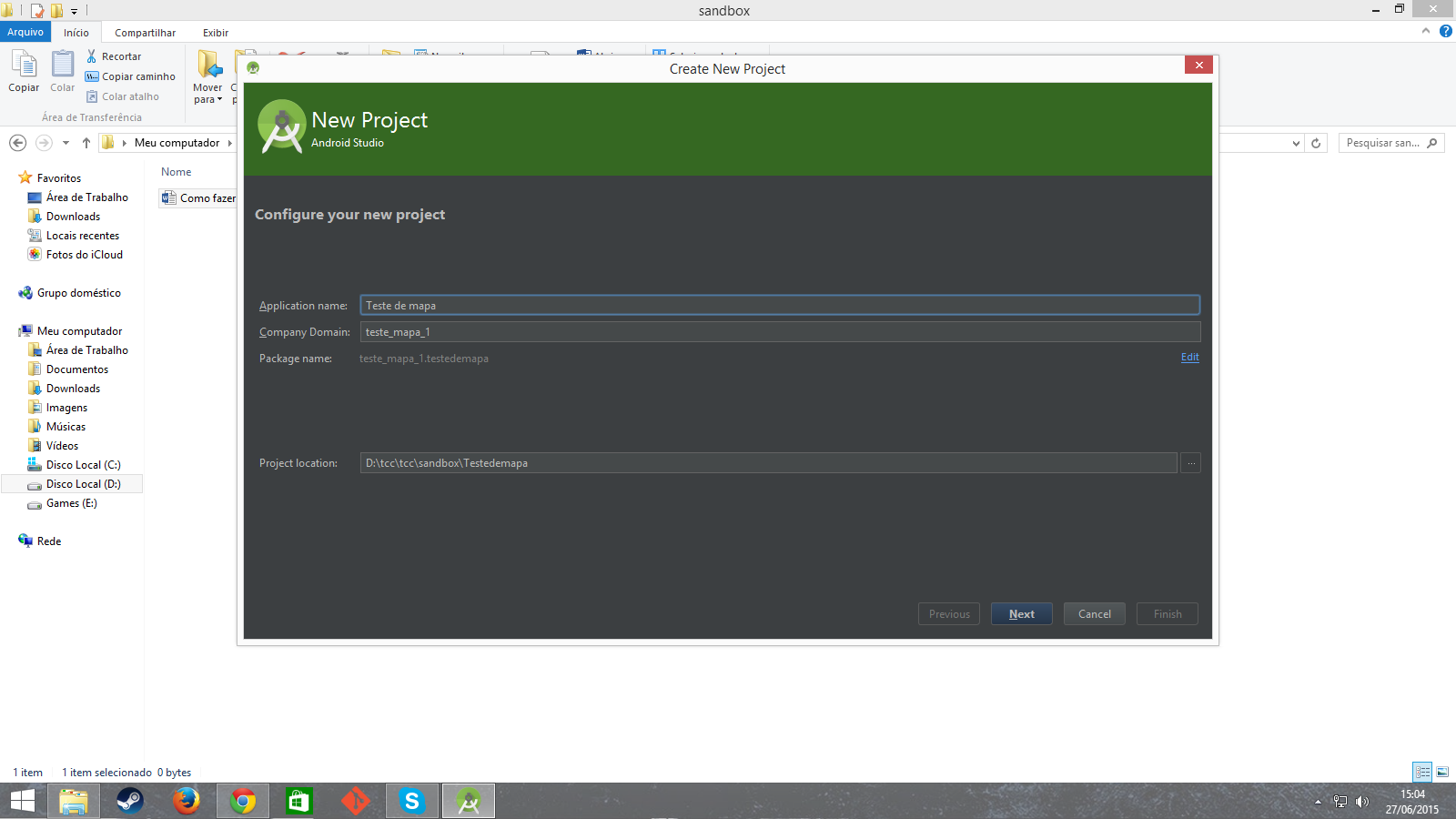
[BIBLIOGRAFIA 29](#_Toc423798970)

# ANDROID STUDIO

O Android Studio é a IDE oficial, oferecido pela Google, para criação de projetos para o sistema operacional Android. O programa tem base o INtelliJ IDEA que oferece soluções para a ferramenta para lidar com variadas aplicações. Para o funcionamento da aplicação é necessário ter o JDK instalado no computador.

A criação e configuração de projetos nessa IDE é bem simplificada, na tela de boas vindas é possivel criar um novo projeto conforme podemos observar na figura 1.

Figura 1: Novo projeto no Android Studio



Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

# EMULADOR

Ao criar aplicações com o Android Studio é necessário realizar testes para a verificação das funcionalidades, mas nem sempre é possível realizar testes direto no smartphone, por esse motivo utilizamos emuladores de sistemas como o genymotion. Com esse software é possível realizar testes do Sistema Operacional Android, tendo suporte para quase todas as funções de um dispositivo móvel.

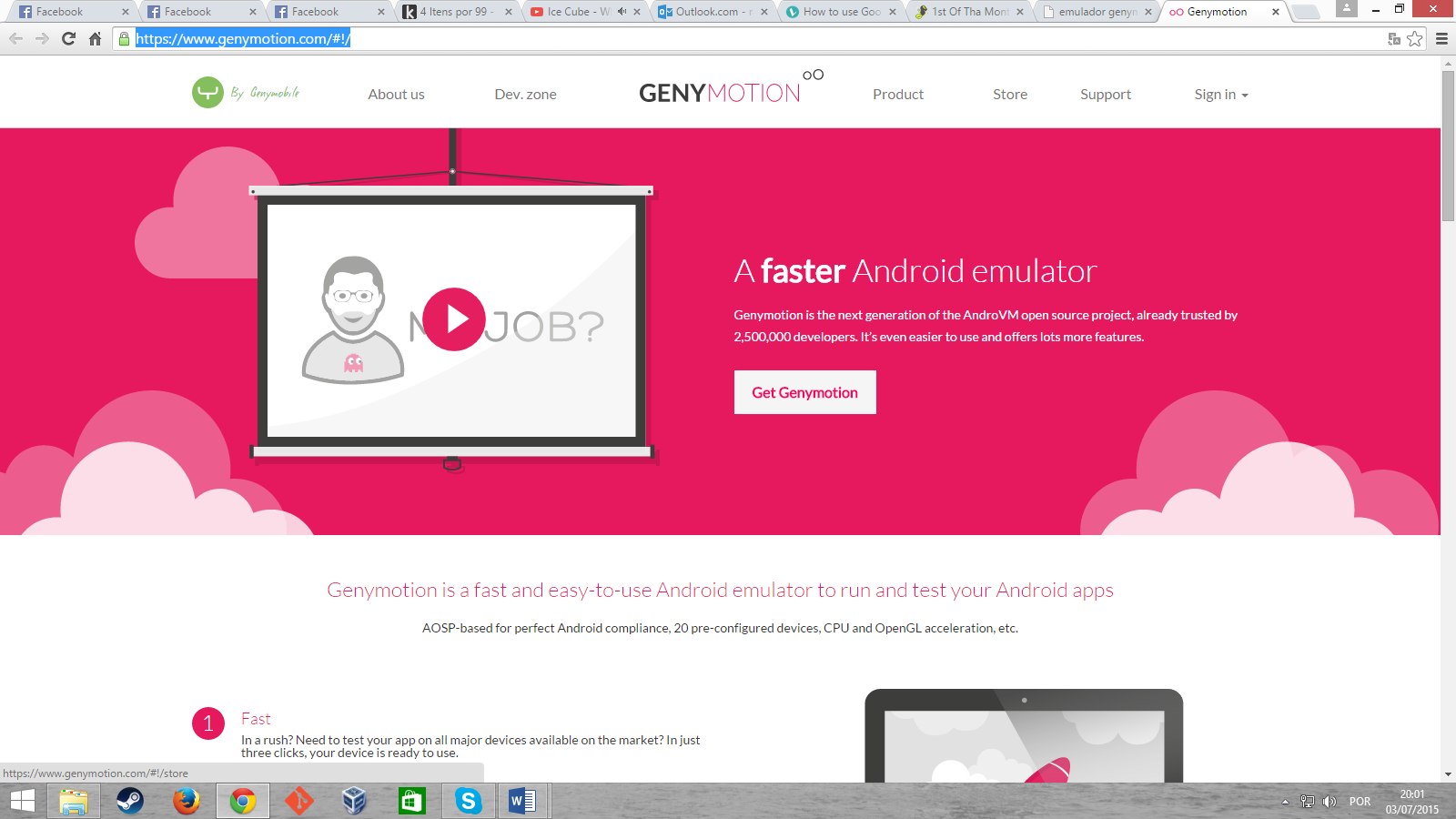
Outro motivo para utilizar um emulador é a velocidade, pois o emulador do próprio android studio é muito lento, além de não suportar processadores da marca AMD, pois esses emuladores precisam de aceleração de hardware e da tecnologia “AMD Virtualization” e processadores AMD com Windows não dão suporte a esse tipo de tecnologia (somente em Linux tem esse tipo de suporte), conforme a documentação diponível no site do developers android.

## INSTALANDO E CONFIGURANDO O EMULADOR GENYMOTION

Para obter o instalador do Genymotion, acesse: [https://www.genymotion.com/#!/](https://www.genymotion.com/" \l "!/)

* Clique em Get Genymotion, conforme figura 2:

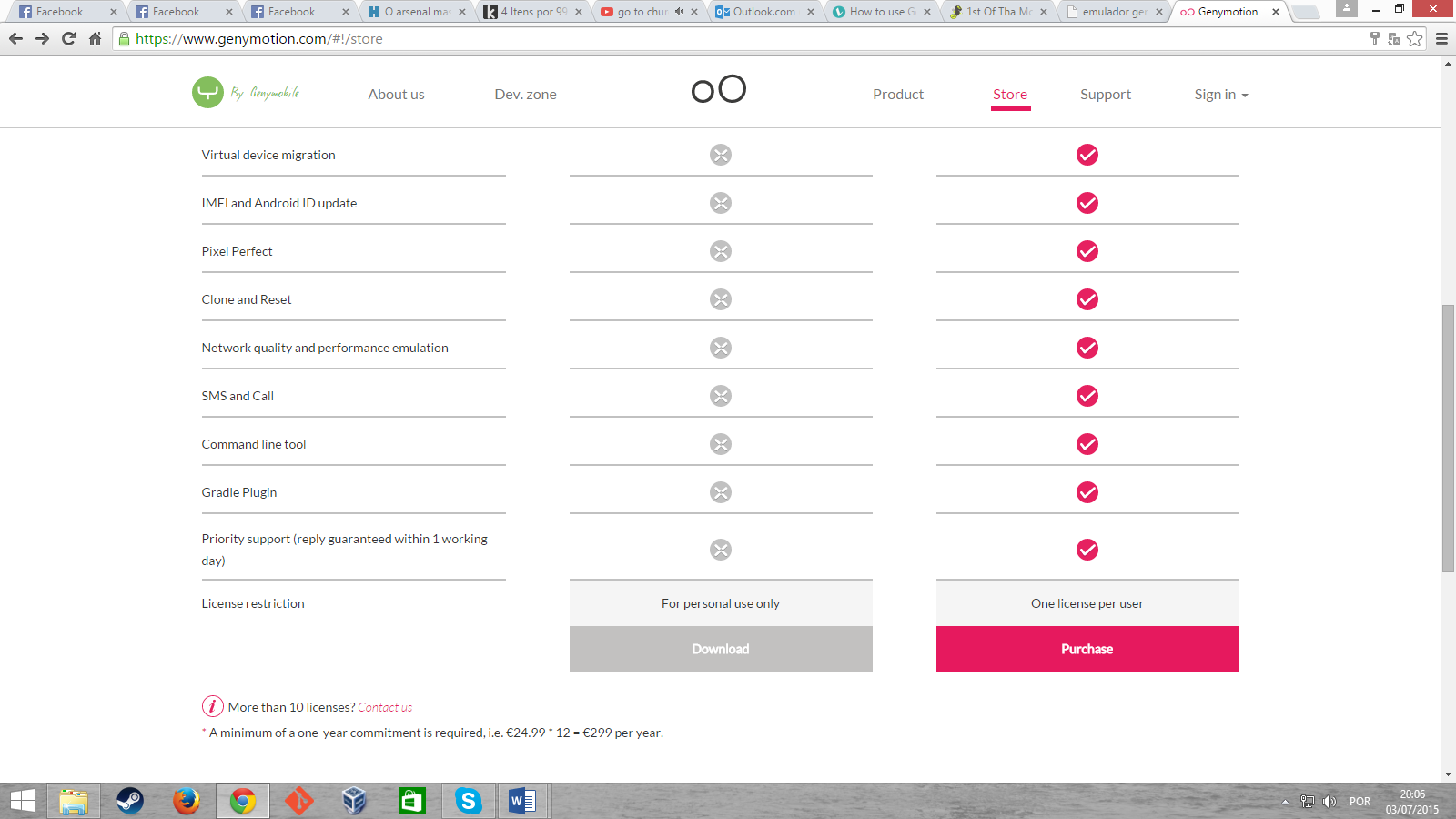
Figura 2: Site do Genymotion



Fonte: Print Screen do Site Genymotion (2015).

* Clicar em Download, para a versão free, conforme figura 3:

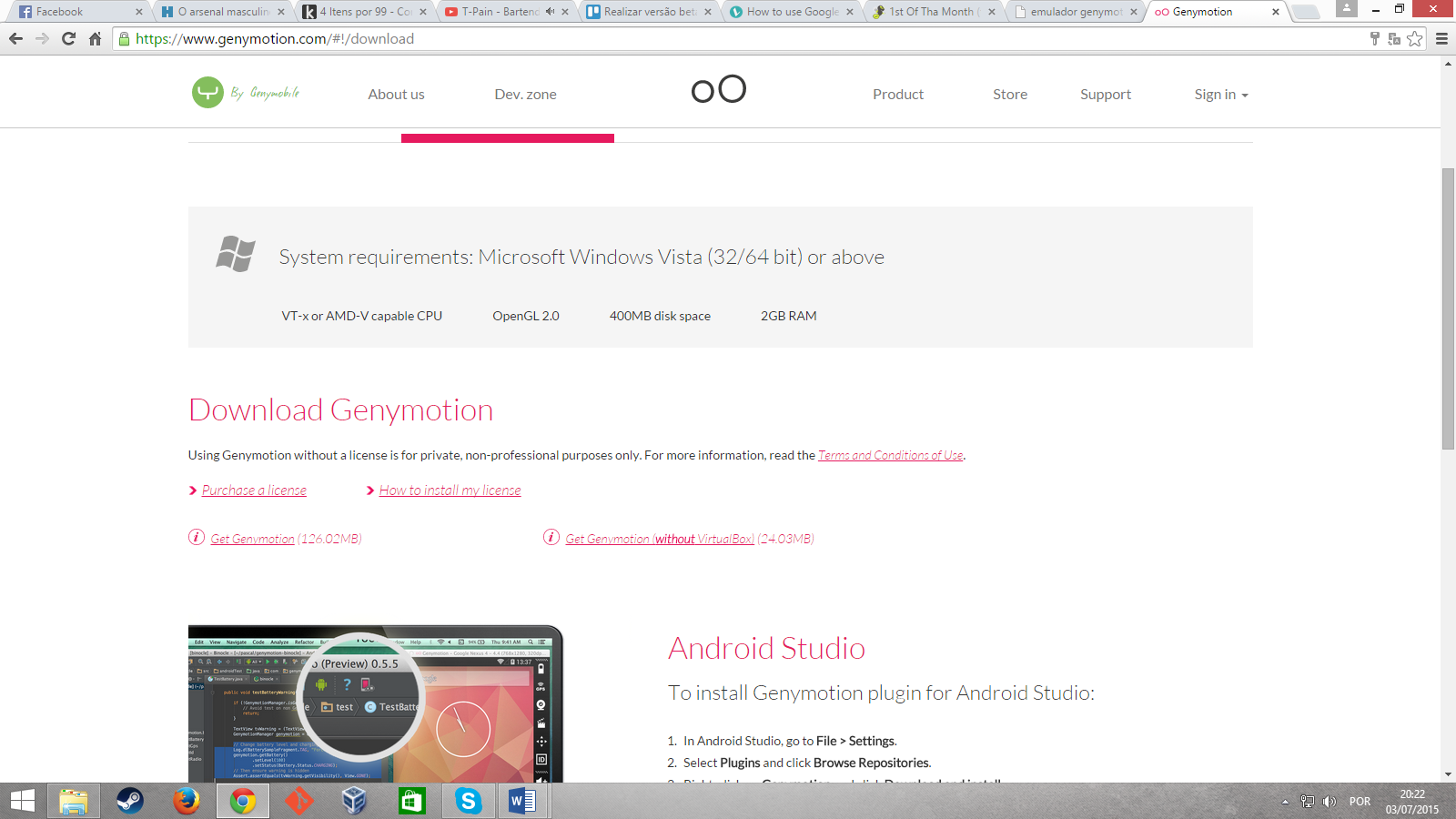
Figura 3: Escolhendo a versao



Fonte: Print Screen do Site Genymotion (2015).

* Escolher a versão do sistema operacional e clicar em Get Genymotion, seguindo a figura 4:

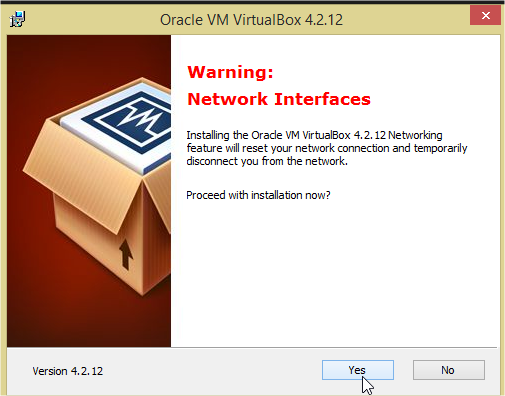
Figura 4: Download Genymotion



Fonte: Print Screen do Site Genymotion (2015).

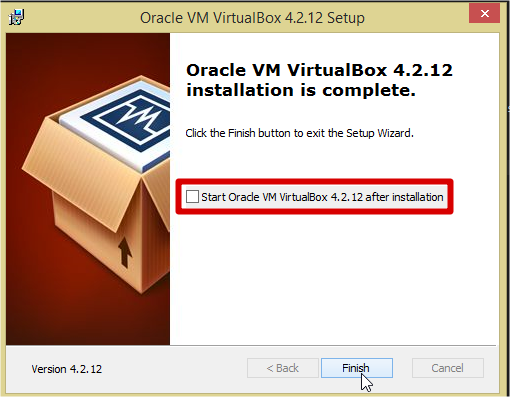
Sera solicitado uma conta no site, caso já tenha basta efetuar seu login. O Genymotion possui duas opções de download uma com a maquina virtual e outra sem para o caso de já houver uma maquina virtual instalada no desktop. Após realizado o download da opção desejada, a instalação estará pronta para ser iniciada. A instalação do software é simples, basta clicar em Yes e Install conforme figura 5 e 6.

Figura 5: Instalação do Genymotion e VM VirualBox



Fonte: Print Screen da janela de instalação do Genymotion (2015).

Figura 6: Instalação do Genymotion e VM VirualBox



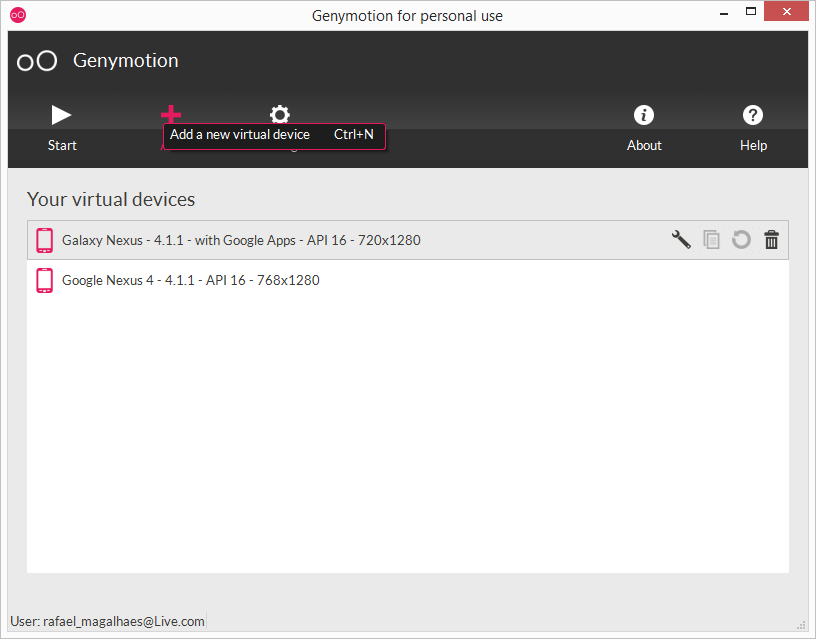
Fonte: Print Screen da janela de instalação do Genymotion (2015).

Após isso a aplicação sera instalada com sucesso.

### Instalando um device

Para instalar um novo device, primeiro é necessário executar o Genymotion. Quando estiver aberto, basta clicar no sinal “+” Add (“Add a new virtual device”), e escolher qual device sera emulado, como é feito na figura 7.

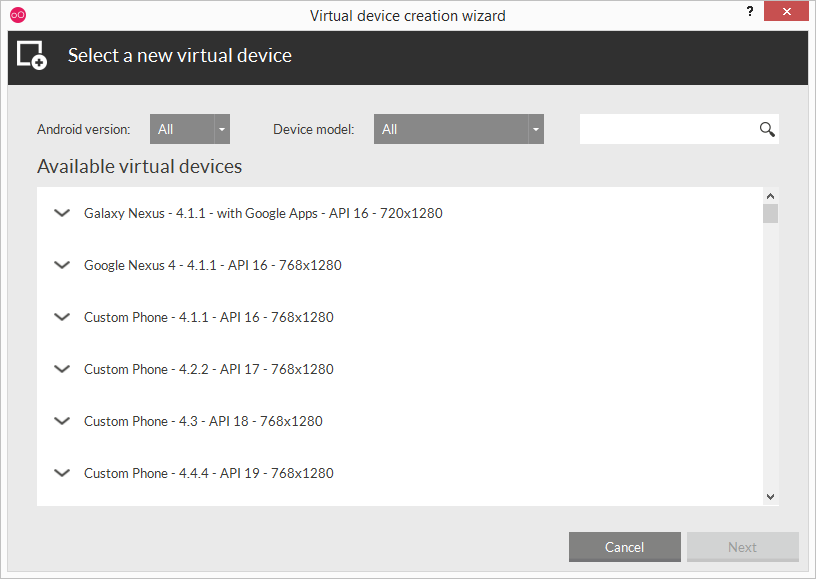
Figura 7: Instalação do Genymotion e VM VirualBox



Fonte: Print Screen da janela de instalação de Devices no Genymotion (2015).

A seguir sera apresentado uma tela com todos os devices disponíveis, como é demonstrado na figura 8. Após a escolha basta clicar em “next” e aguardar o processo para que se possa ter um device emulado da versão do android que foi escolhida.

Figura 8: Escolhendo devices



Fonte: Print Screen da janela de instalação de Devices no Genymotion (2015).

## CONFIGURAR EMULADOR COM O GOOGLE PLAY STORE

Após instalado um device, ainda não se pode testar aplicações que dependam de recursos do google play services como o serviço de mapas. A instalação desse recurso se da manualmente. Nesse capitulo sera documentado como é realizado o procedimento.

Os passos são simples, antes de tudo será necessário fechar todos os processos do Genymotion (basta fechar a aplicação, se a mesma estiver aberta). E efetuar o download dos algunss arquivos, que correspondam as caracteristicas dos devices selecionados:

Android 4.1:

* genymotion\_vbox86p\_4.1.1\_130923\_155330\_w\_gapps.ova

<https://drive.google.com/file/d/0B_OTYoqMpv2ES2lsa3ZjOE4tYXM/edit>

* .bin

<https://drive.google.com/file/d/0B_OTYoqMpv2EMkQ2MlAyR0tkT2M/edit>

Link Alternativo:

Android 4.2:

* Galaxy S4 HTC One Xperia Z 4.2.2 with Google Apps API 17:

<http://www.mediafire.com/download/jdn83v1v3bregyu/Galaxy+S4++HTC+One++Xperia+Z+-+4.2.2+-+with+Google+Apps+-+API+17+-+1080x1920.zip>

Utilizando os dois primeiros links os passos seguintes serão:

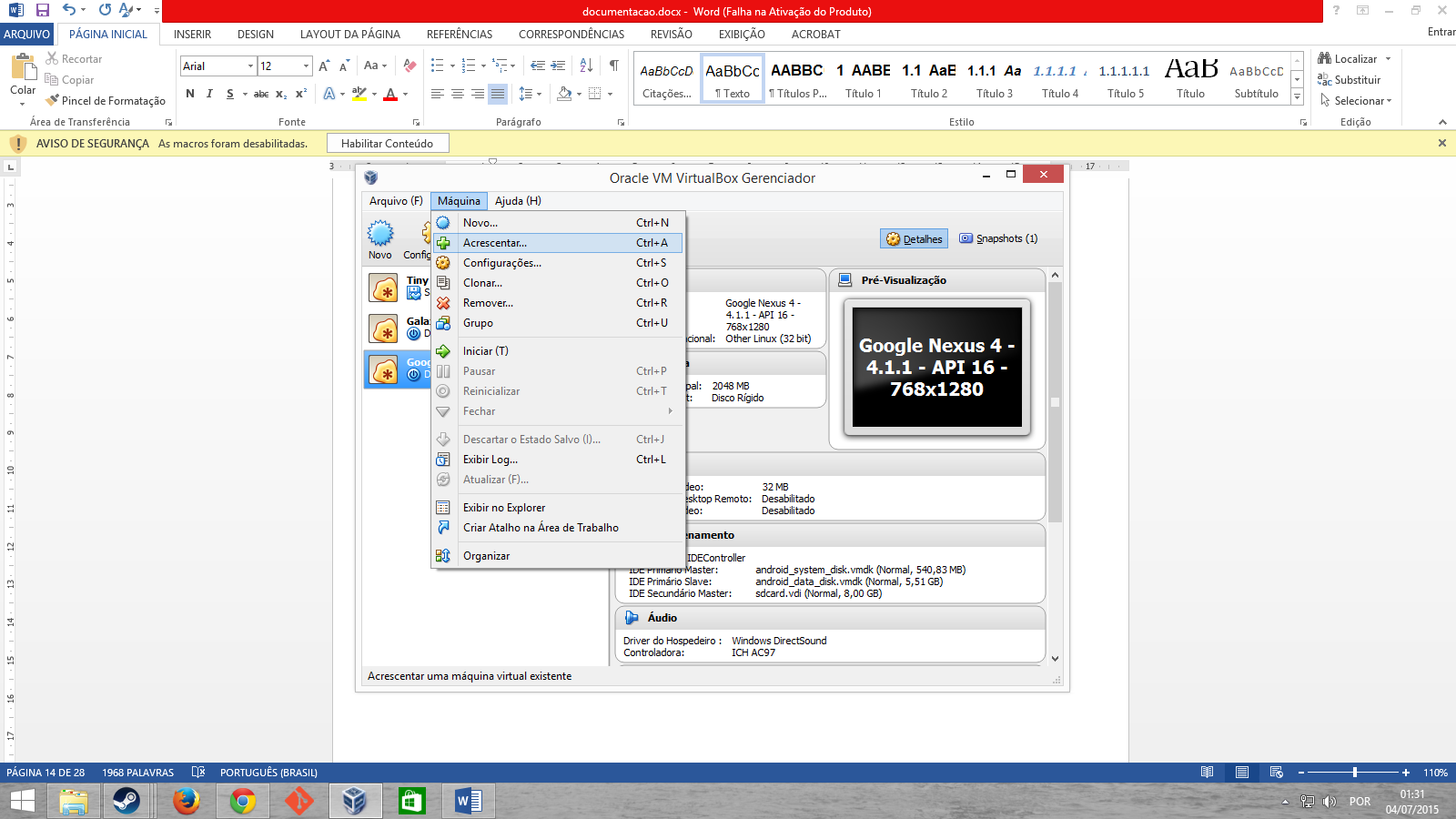
* Localizar a pasta Genymotion dentro da pasta Genymobile que fica no caminho: C:\Users\Nome\_do\_seu\_Usuario\AppData\Local\Genymobile\Genymotion
* Copiar o arquivo com extensão “.ova” que foi feito o download para dentro da pasta ova (caso tenha outros arquivos .ova dentro da pasta, exclua todos): C:\Users\Nome\_do\_seu\_Usuario\AppData\Local\Genymobile\Genymotion\ova
* Copiar o arquivo .bin que foi feito o download para dentro da pasta templates (caso tenha outros arquivos .bin dentro da pasta, exclua todos): C:\Users\Nome\_do\_seu\_Usuario\AppData\Local\Genymobile\Genymotion\templates
* Abrir o Genymotion e criar um novo device, escolhendo a versão do Android 4.1.1. Agora podemos observar que tem disponível um novo device com a seguinte nomenclatura: Galaxy Nexus – 4.1.1 – with Google Apps – API 16 – 720X1280, crie o device em cima dessa opção.

Após criado, pode-se executa-lo e com o emulador já aberto se cadastrar na google play com seu email e senha. Caso solicite alguma atualização, faça todas (exceto do Genymotion que sera solicitado em toda execução desse device).

O procedimento para o link alternativo é mais rápido, basta realizar o dowload dos arquivos e:

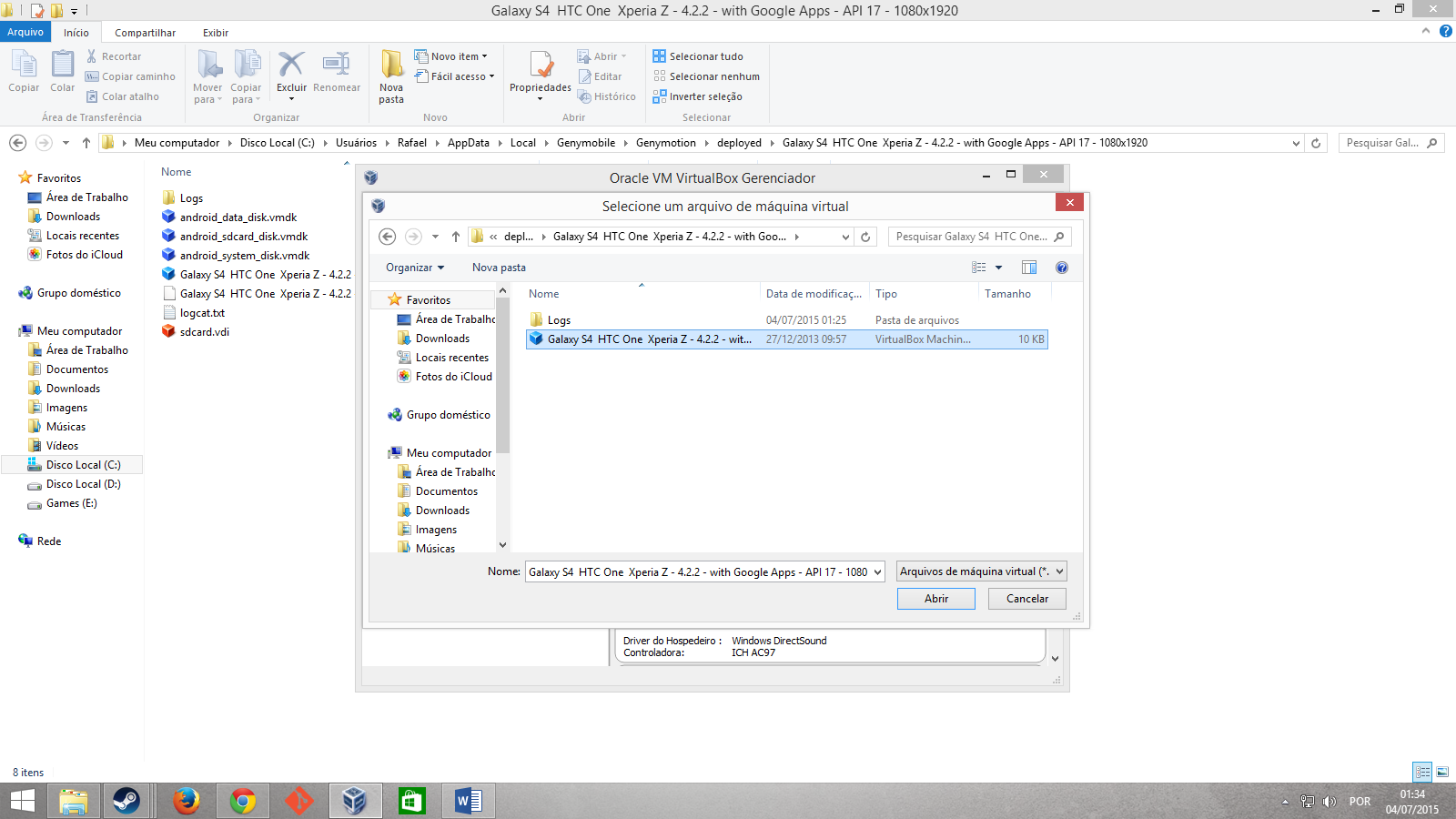
* Extrair os arquivos
* Copiar a pasta extraída para o caminho (atenção para a localização da pasta deployed):C:\Users\Nome\_do\_seu\_usuario\AppData\Local\Genymobile\Genymotion\Genymobile\Genymotion\deployed
* Abrir o VM VirtualBox e no menu Maquina, escolher Acrescentar.. e localizar a pasta em que foi copiado os arquivos e acrescentar o arquivo clicando em abrir , conforme figura 9 e 10: C:\Users\Rafael\AppData\Local\Genymobile\Genymotion\deployed\Galaxy S4 HTC One Xperia Z - 4.2.2 - with Google Apps - API 17 - 1080x1920

Figura 9: Acrescentando novo device



Fonte: Print Screen da janela de instalação de Devices no VirtualBox (2015)

Figura 10: Acrescentando novo device



Fonte: Print Screen da janela de instalação de Devices no VrtualBox (2015)

Agora o software possui dois devices diferentes com o google play instalado.

# API DE MAPAS COM O ANDROID STUDIO

A API de mapas do Google permite que o desenvolvedor insira na criação da sua aplicação mapas fornecidos pela empresa. A implementação da funcionalidade no código fonte é simples, para quem já tem conhecimento em linguagem de programação. Nesse capitulo iremos explicar como funciona o código fonte e como é feito o procedimento para inserir mapas em aplicações Android.

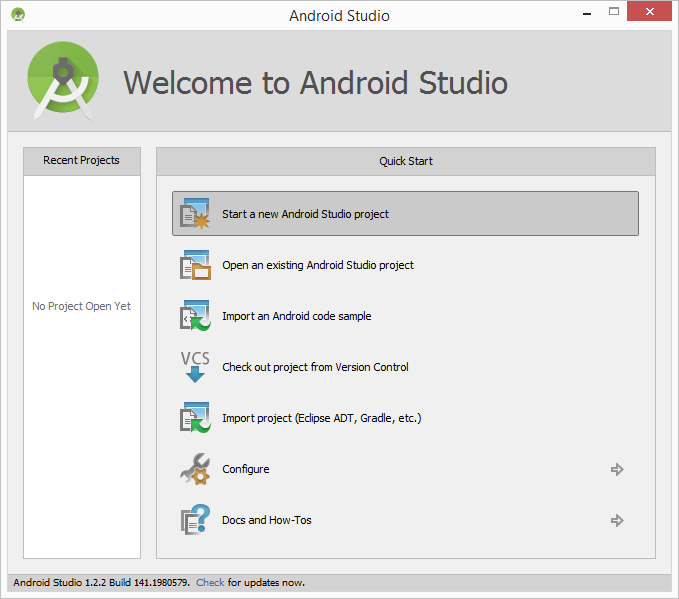
Todo o procedimento foi feito em uma máquina com a seguinte configuração:

* Processador: AMD PHENOM II X4
* Memoria: 8GB
* HD: 1TB
* Sistema operacional: Windows 8.1 Pro

## Começando um novo projeto

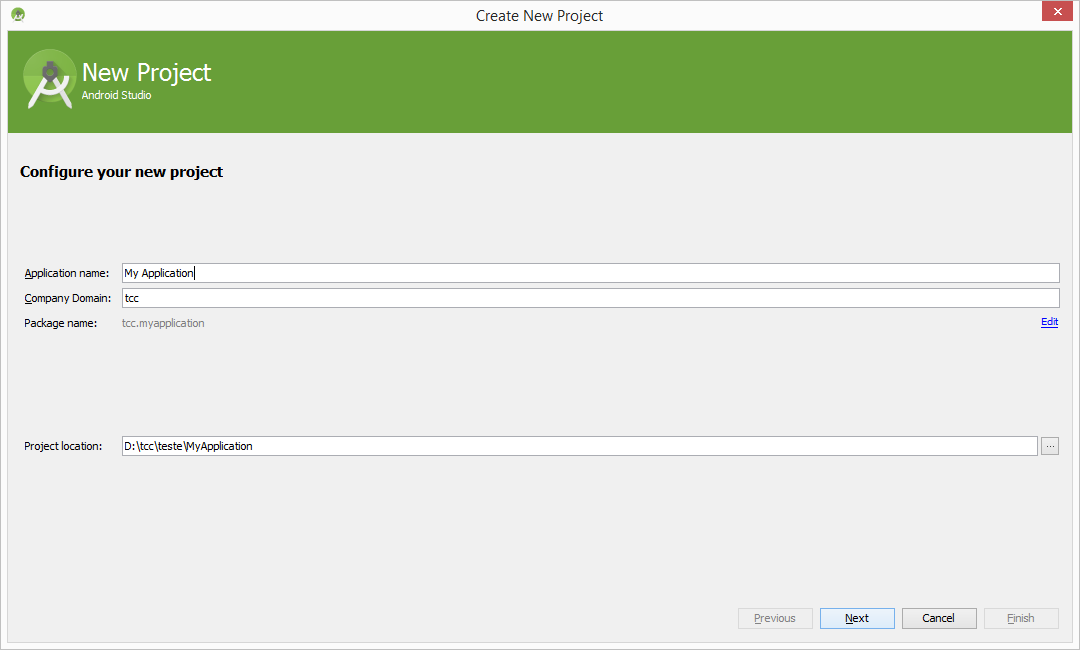
A criação e configuração de projetos nessa IDE é bem simplificada, na tela de boas vindas é possivel criar um novo projeto conforme podemos observar na figura 11. O primeiro passo para iniciar o projeto é cria-lo conforme a figura 12 e configurar para que o aplicativo seja suportado no mínimo na versão 4.1 (JellyBean) do Android para que as funcionalidades inseridas funcionem sem problemas ou possíveis erros, de acordo com a figura 13.

Figura 11: Começando um novo projeto



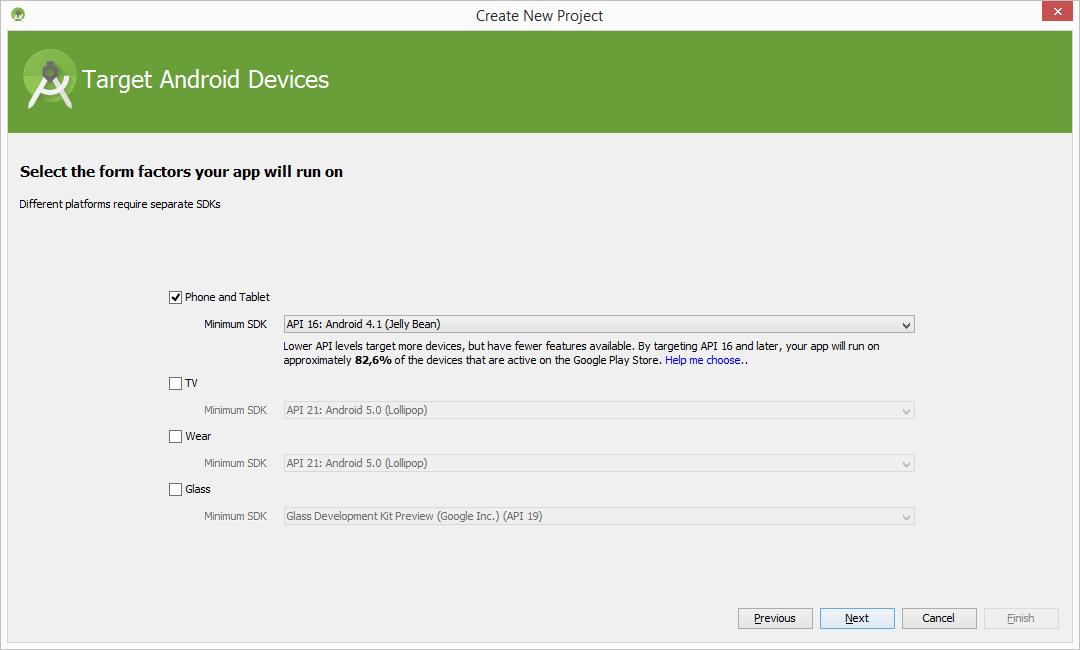
Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

Figura 12: Novo projeto no Android Studio



Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

Figura 13: Escolhendo o requisito minimo para o SDK



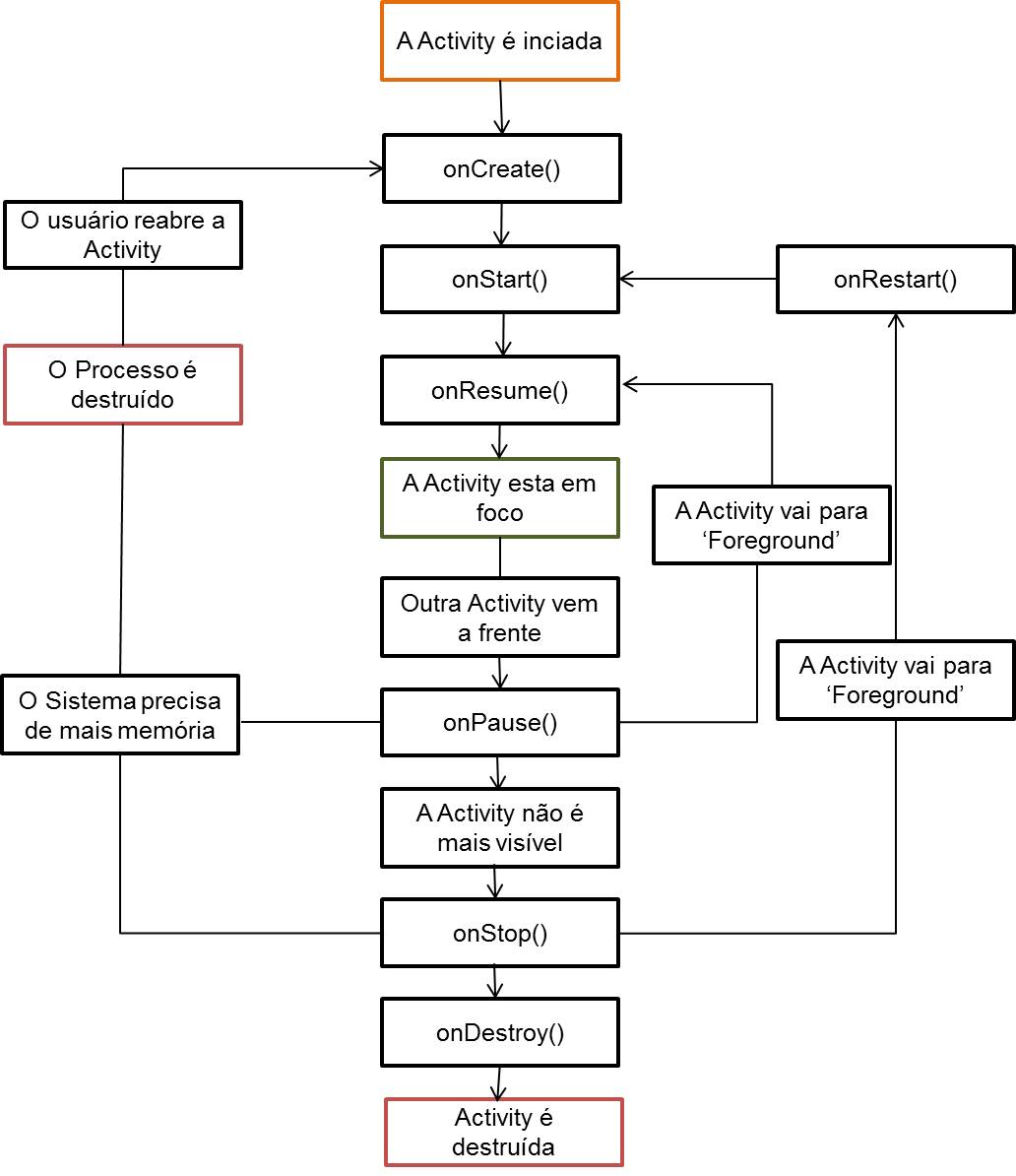
Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

## Acitivity

A próxima etapa é a escolha de uma activity. Conforme Silveira (2010), activity é uma classe do sistema Android que gerencia a UI[[1]](#footnote-1), e todo aplicativo android começa a partir de uma activity.

Uma maneira de fácil entendimento de uma activity é ver como é seu funcionamento, ou ciclo de vida, como podemos observar na figura 14.

Figura 14: Ciclo de Vida de uma Activity



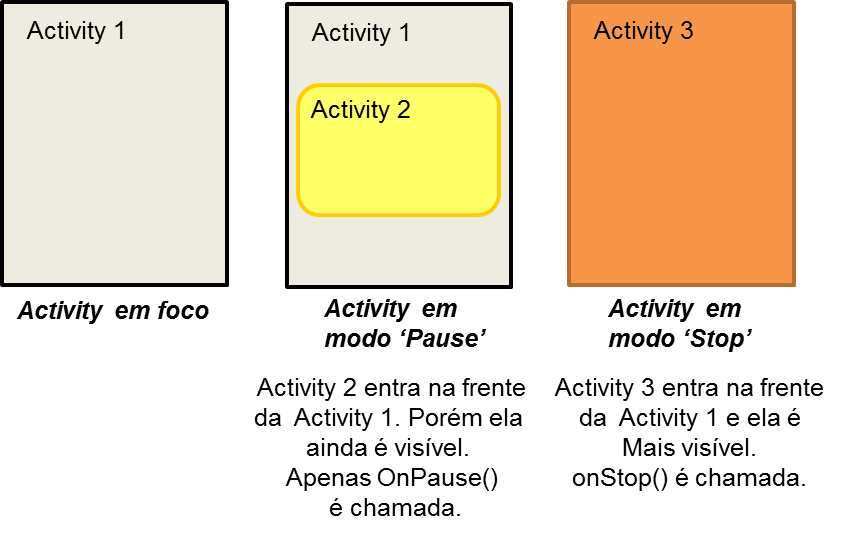
Fonte: Adaptado de Silveira (2010) pelos autores.

Conforme o site da Android Developers as funções de um clico de vida da activity pode ser definida por:

* onCreate() Primeira função executada pela Activity. Normalmente é a responsável por carregar layouts XML e outras funções de inicialização. É executada somente uma vez durante a vida da Activity.
* onStart() É executada após a onCreate() e também quando uma Activity que estava em sendo executada em background e volta a ter foco.
* onResume() tem quase a mesma função da onStart(), é chamada na inicialização da Activity (após a onStart()) e também quando uma Activity volta a ter foco. A diferença da onStart() é que ela só é chamada quando a Activity não estava mais visível na tela e volta a ter o foco, enquanto a onResume() sempre é invocada nas retomadas de foco.
* onPause() Primeira função a ser chamada quando a Activity perde o foco.
* onStop() – é chamada quando a Activity fica completamente encoberta por outra Activity ou seja não é mais visível.
* onDestroy() A última função a ser executada. Sendo que depois da execução a activity nao pode mais ser relançada. Se o usuário voltar a requisitar essa Activity, outro Activity sera construida.
* onRestart() invocada antes da onStart(), quando uma Activity volta a ter o foco depois de estar em background.

Para completar o entendimento de como é uma activity podemos definir seus estados definidos pela figura 15.

Figura 15: Focos da activity

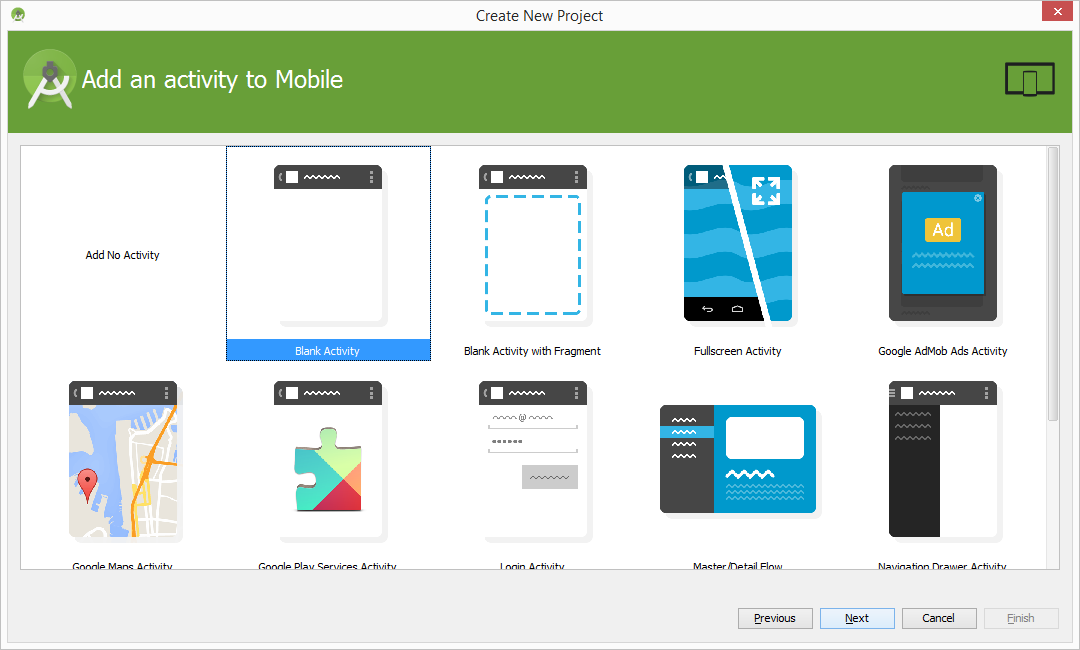


Fonte: Adaptado de Silveira (2010) pelos autores.

## Escolha de um Activity

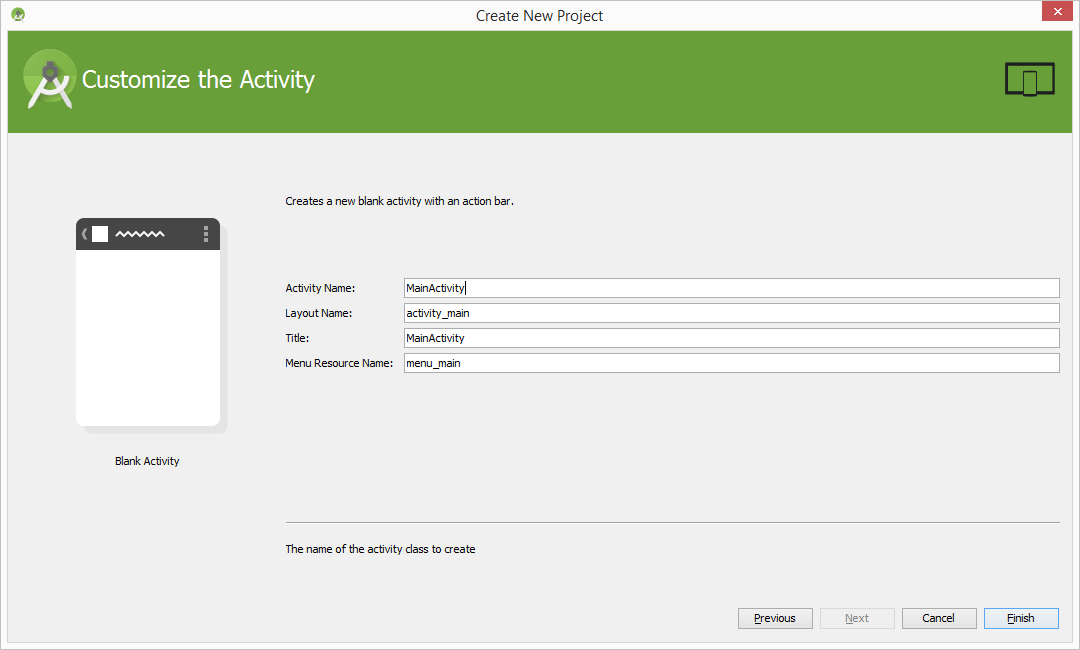
Seguindo com a criação de um projeto com a API de mapas, após a configuração inicial do seu projeto será necessário a escolha inicial da activity, escolhemos a “blank activity” por ser uma aplicação basica para programação, a partir disso e preciso definir um nome para a activity e para o layout conforme figura 16 e 17.

Figura 16: Escolha de um activity



Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

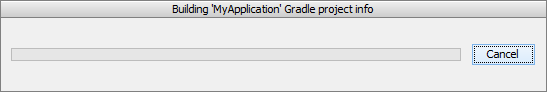
Figura 17: Escolhendo um nome para a Activity



Fonte: Print Screen da janela de Criação de Novo Projeto (2015).

Após a configuração inicial, o processo para a inicialização do projeto é iniciado, visto da maneira da figura 18.

Figura 18: Carregando um projeto

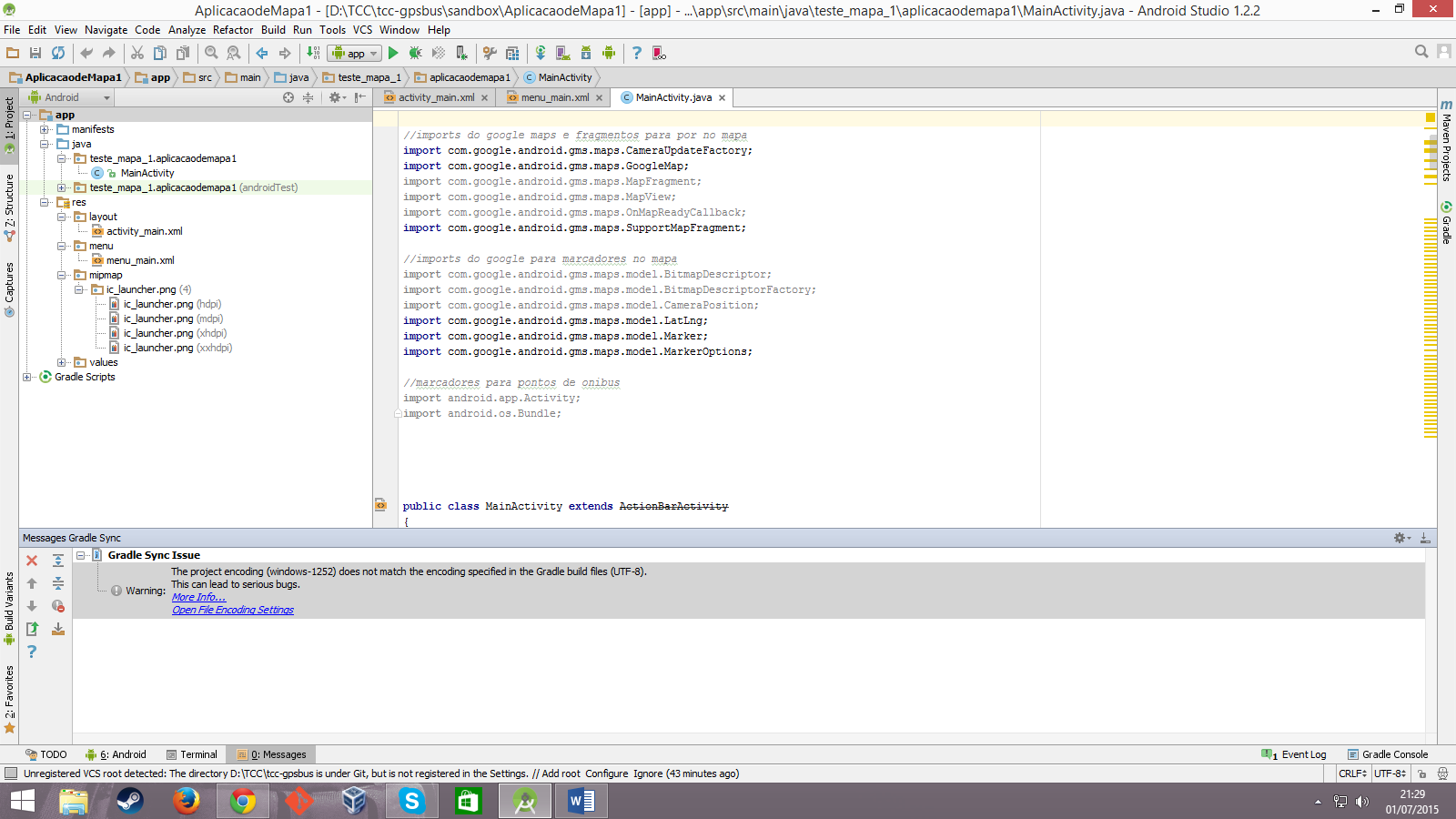


Fonte: Print Screen da janela de carregamento de um novo projeto (2015).

## Iniciando o projeto

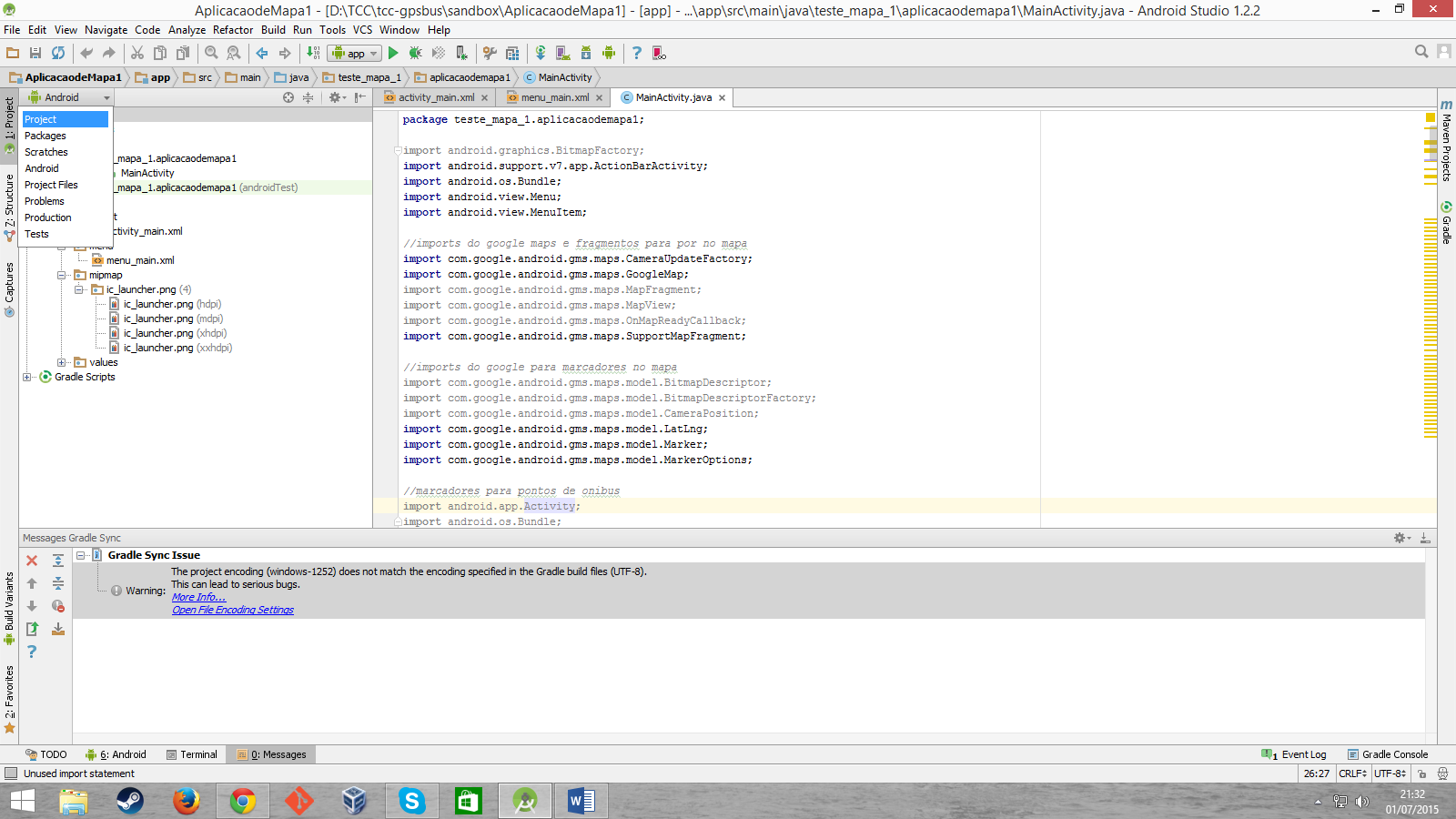
Com a criação do projeto pronta, a primeira tela que será exibida é a da figura 19, e a primeira configuração que necessita ser alterada no layout da IDE é a disposição dos arquivos, iremos mudar para “Project”, a figura 20 demonstra onde fica a alteração.

Figura 19: Tela principal do projeto



Fonte: Print Screen da tela principal do projeto (2015).

Figura 20: Alterando a disposição das pastas



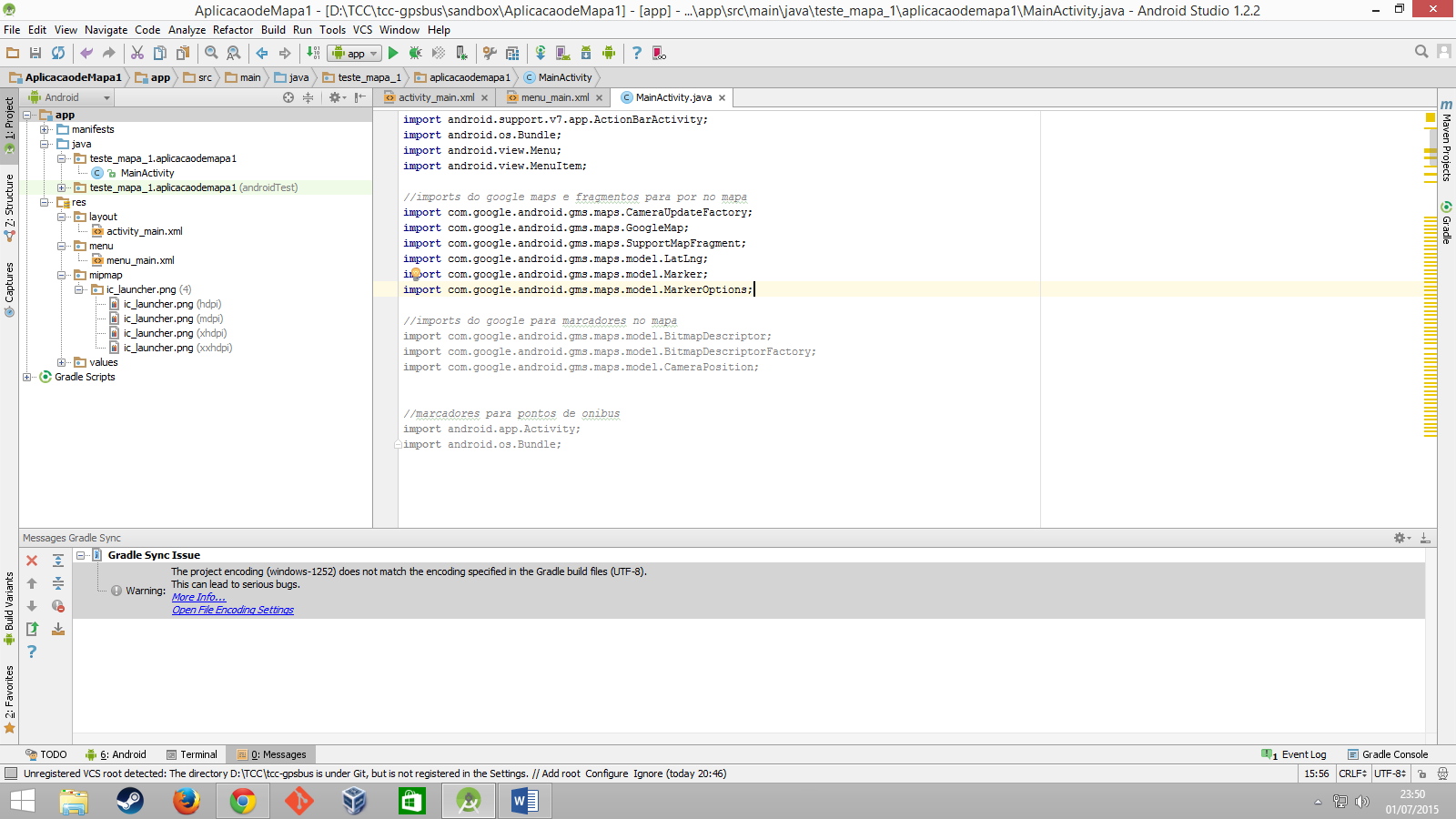
Fonte: Print Screen de alteração da disposição de pastas (2015).

## Imports

O procedimento a ser seguido agora é importar a biblioteca de marcadores e de mapas do google. Na arvore Java > Nomedoprojeto > MainActivity deve-se inserir algumas importações, a figura 21 exemplifica onde sera inserido o código abaixo:

* **Import** com.google.android.gms.maps.CameraUpdateFactory;
* **Import** com.google.android.gms.maps.GoogleMap;
* **import** com.google.android.gms.maps.SupportMapFragment;
* **Import** com.google.android.gms.maps.model.LatLng;
* **Import** com.google.android.gms.maps.model.Marker;
* **import** com.google.android.gms.maps.model.MarkerOptions;

Figura 21: Inserindo imports



Fonte: Print Screen de inserção de bibliotecas (2015).

De acordo com o site Google developers essas importações são necessárias para que no código fonte possa ser acessado as classes de cada pacote, não sendo necessário digitar o nome de cada pacote toda vez que referenciar uma classe.

Cada import tem sua funcionalidade:

* com.google.android.gms.maps.CameraUpdateFactory;

Essa importação serve para criar classes que tenham métodos CameraUpdate() que são usados para modificação da câmera do mapa.

* com.google.android.gms.maps.GoogleMap;

Essa é importação da classe principal da API Google Maps, ela contém todos os métodos relacionados ao mapa.

* com.google.android.gms.maps.SupportMapFragment;

É um componente do mapa na aplicação. Faz com que o mapa seja inserido na aplicação tratando do seu ciclo de vida.

* com.google.android.gms.maps.model.LatLng;

Classe que representa coordenadas (Latitude, Longitude) é representada em graus.

* com.google.android.gms.maps.model.Marker;

Trata-se de uma classe que possibilita a inserção de ícones em pontos na superfície do mapa.

* com.google.android.gms.maps.MakerOptions

Define opções de marcação para a classe Marker.

## Variáveis

Após a definição das importações, será necessário inicializar variáveis. Variável em computação é uma região da memoria do computador que tem a finalidade de armazenar informações de um programa por um tempo determinado. As variáveis que serão iniciadas serão:

private GoogleMap mapa;

* Private: define que a variável não pode ser acessada por outra subclasse.
* GoogleMap: instanciação da classe para mapas.
* mapa: nome da variável (pode ser qualquer outro nome).

Com essa parte do código pronta, executando o projeto já temos uma aplicação funcionando com o mapa do google.

## Alterações na Interface

É preciso que haja uma alteração do arquivo XML, mais precisamente o “activity\_main.xml”, esse arquivo fica responsável em tratar da parte da interface, permitindo que o desenvolvedor defina a UI separadamente, não interferindo na funcionalidade da aplicação. O acesso ao XML desse arquivo fica na arvore: res > layout > activity\_main.xml, nesse arquivo será necessário acrescentar um fragmento para mostrar o mapa na tela do smartphone. “Fragment” de acordo com o site Developers Android é uma seção modular de uma acitivity, tendo seu próprio ciclo de vida, recebendo também seus próprios eventos de entrada.

<**fragment android:id="@+id/mapa"**

**android:layout\_width="match\_parent"**

**android:layout\_height="match\_parent"**

**class="com.google.android.gms.maps.SupportMapFragment"** />

* android:id=”@+id/mapa”: nome da id para criar uma nova entrada no R.java
* android:layout\_width=”match\_parent”: define o tamanho que o layout ira ocupar na tela
* android:layout\_height=”match\_parent”: tambem define o tamanho que o layout ira ocupar na tela
* Class: classe a ser usada pelo fragment.

Retornando a MainActivity, sera inserido a linha de comando

**mapa** = ((SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.***mapa***)).getMap();

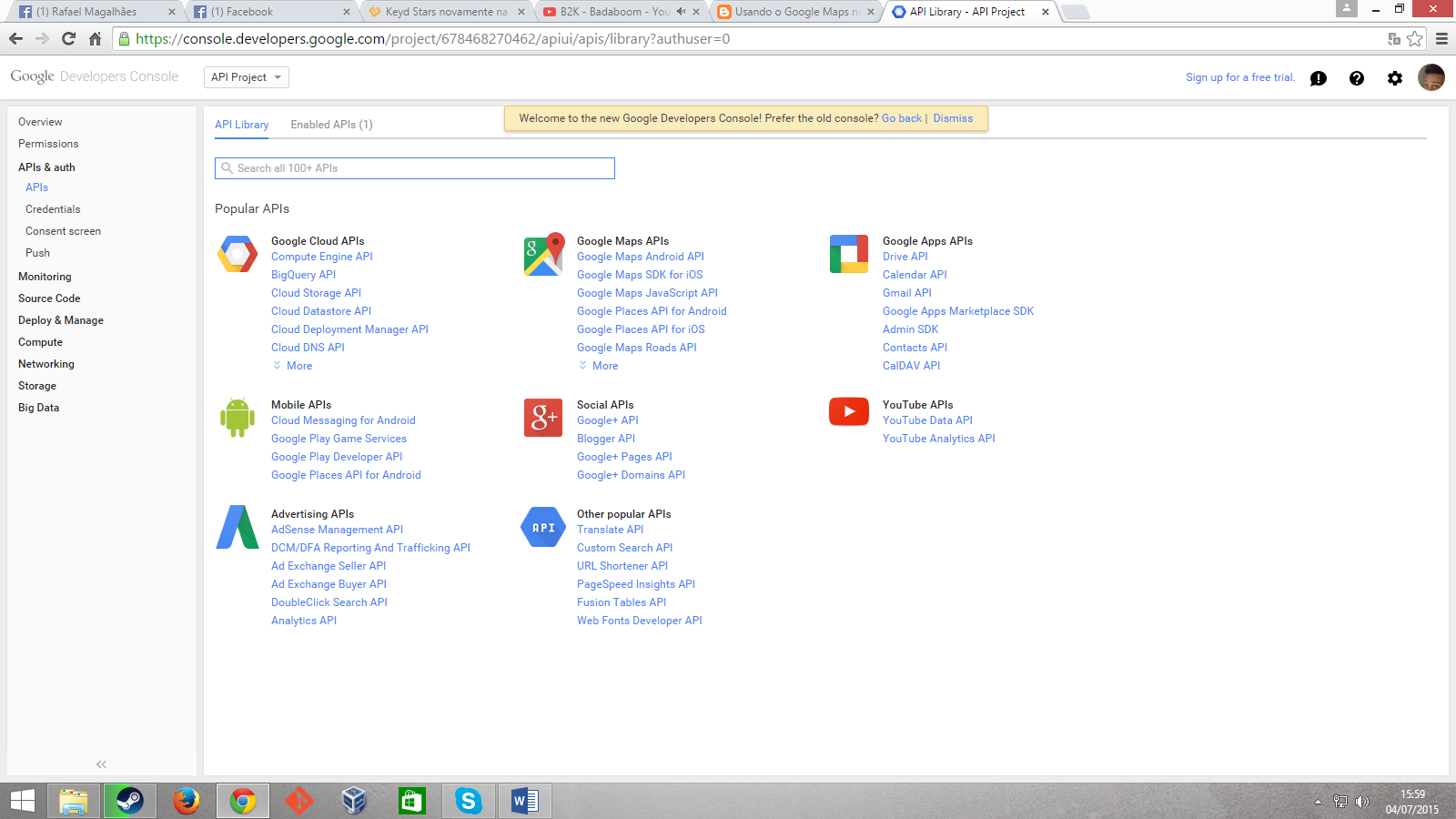
* findFragmentByID: interface para interagir com o fragment da activity
* getSupportFragmentManaget: retorna valores de interação de fragments com essa activity.

### API KEY

Para utilizar qualquer API do Google, é preciso ter uma API key[[2]](#footnote-2). API Key é um código criado em programas de computador chamando uma API para identificar o programa e o desenvolvedor. São usadas para monitorar e controlar a forma como a API esta sendo usada.

Para obter uma Api key para o uso de mapas da Google, é preciso entrar no site: <https://console.developers.google.com/project/678468270462/apiui/apis/library?authuser=0>, procurar pela API correspondente, nesse caso Google Maps Android API. A tela que sera apresentada para o usuário é a a figura 22.

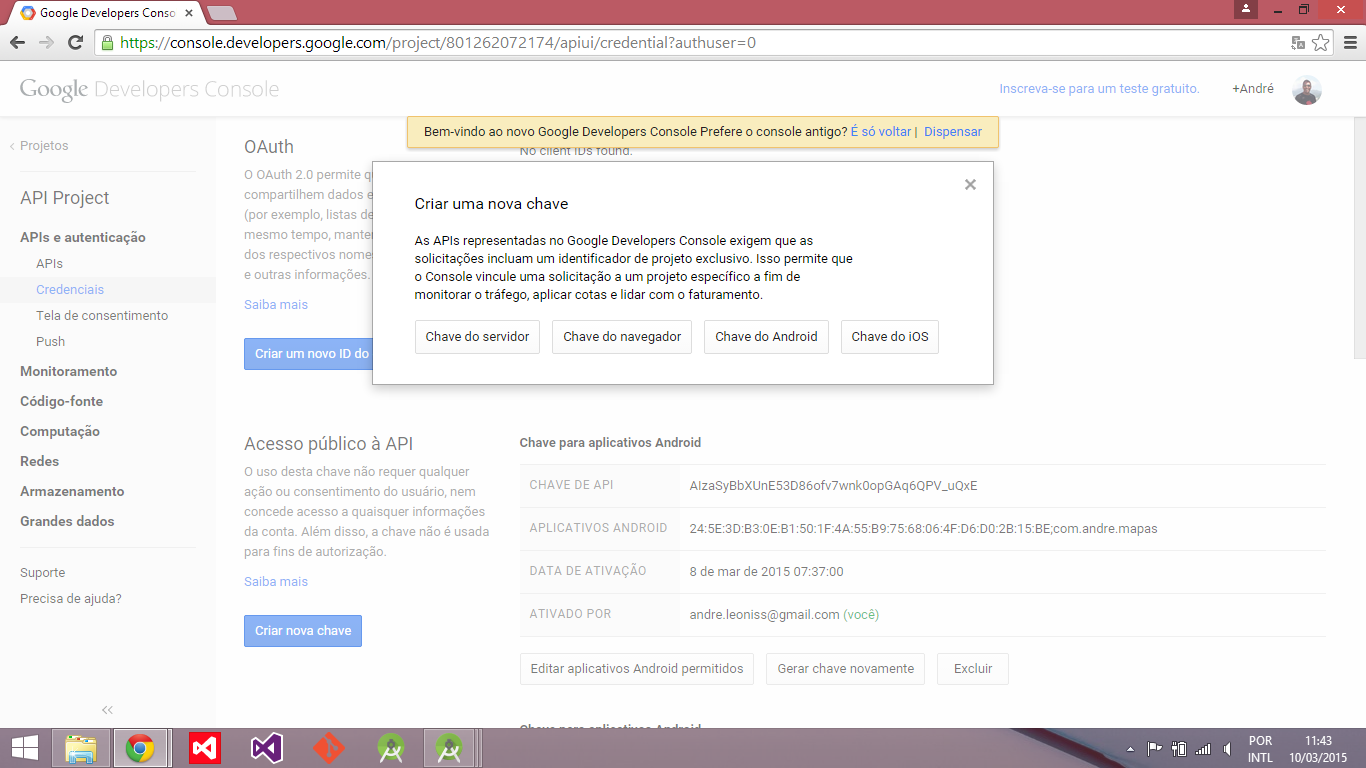
Figura 22: Api Key



Fonte: Print Screen da página onde se encontram API Key (2015).

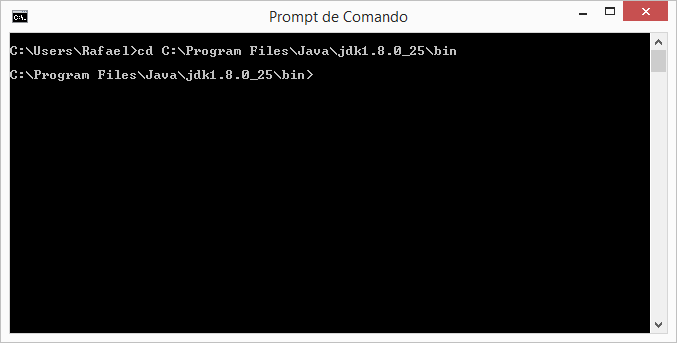
Após localizado a API do Google maps para Android, deve-se clicar em criar nova chave, e depois em chave do Android, conforme a figura 23.

Figura 23: Api Key



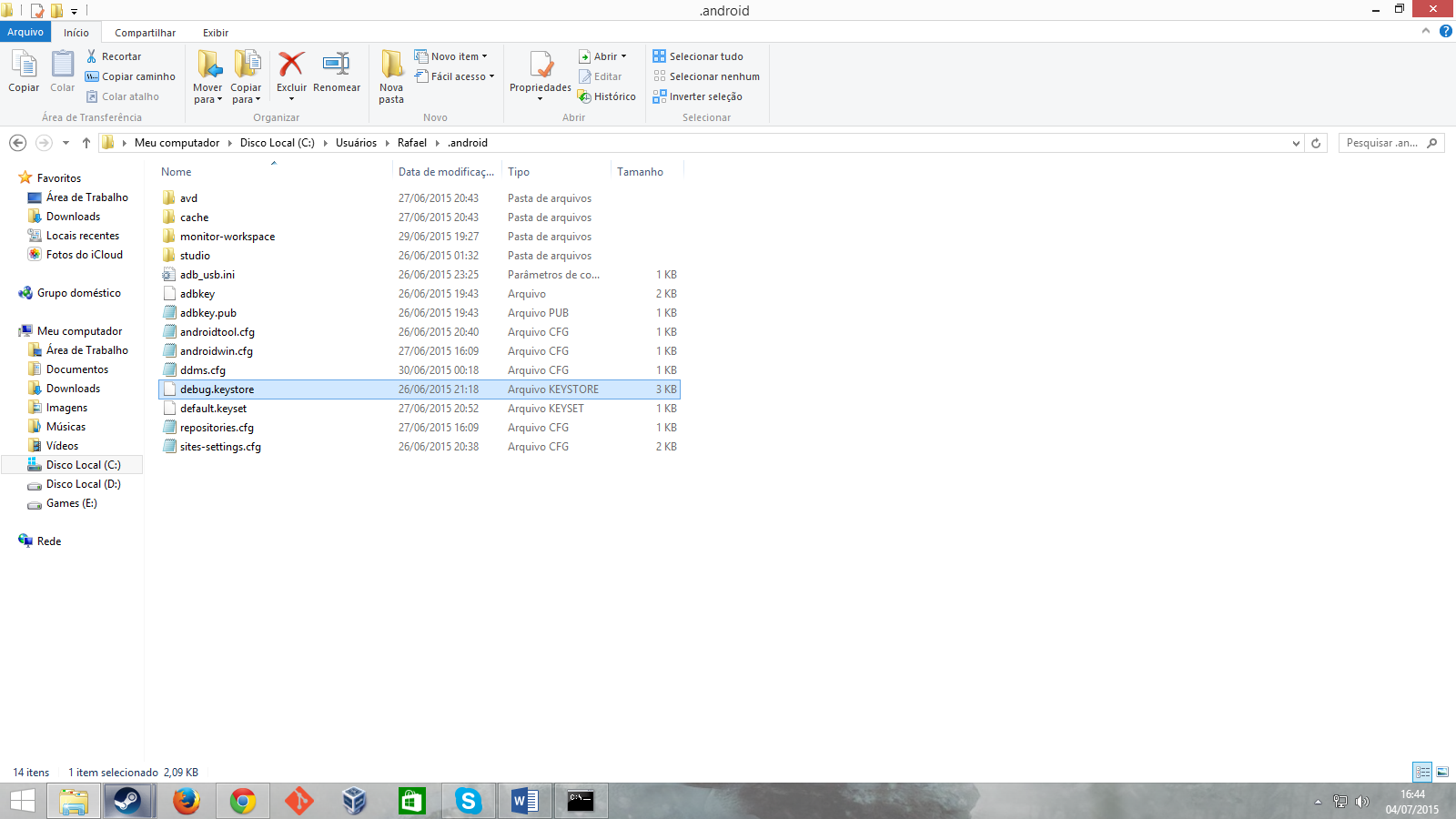
Fonte: Print Screen da página onde se encontram API Key (2015).

A proxima etapa é obter uma chave para ser implantada no aplicativo android, para isso sera aberto o cmd do Windows e localizado a pasta bin do java, que normalmente esta nesse caminho: C:\Program Files\Java\jdk1.8.0\_25\bin, na pasta bin do Java deve ser digitado a linha de comando: keytool -list -v –keystore seguido pelo caminho do debug.keystore, que geralmente se encontra: C:\Users\Rafael\.android\debug.keystore, aperte enter e sera solicitado uma senha, como ela não foi definida previamente, basta apertar enter novamente para obter a chave SHA1, como nas imagens 24, 25 e 26.

Figura 24: Api Key

Fonte: Print Screen da janela do Prompt de Comando (2015).

Figura 25: Api Key



Fonte: Print Screen da janela do diretório onde se encontra a chave (2015).

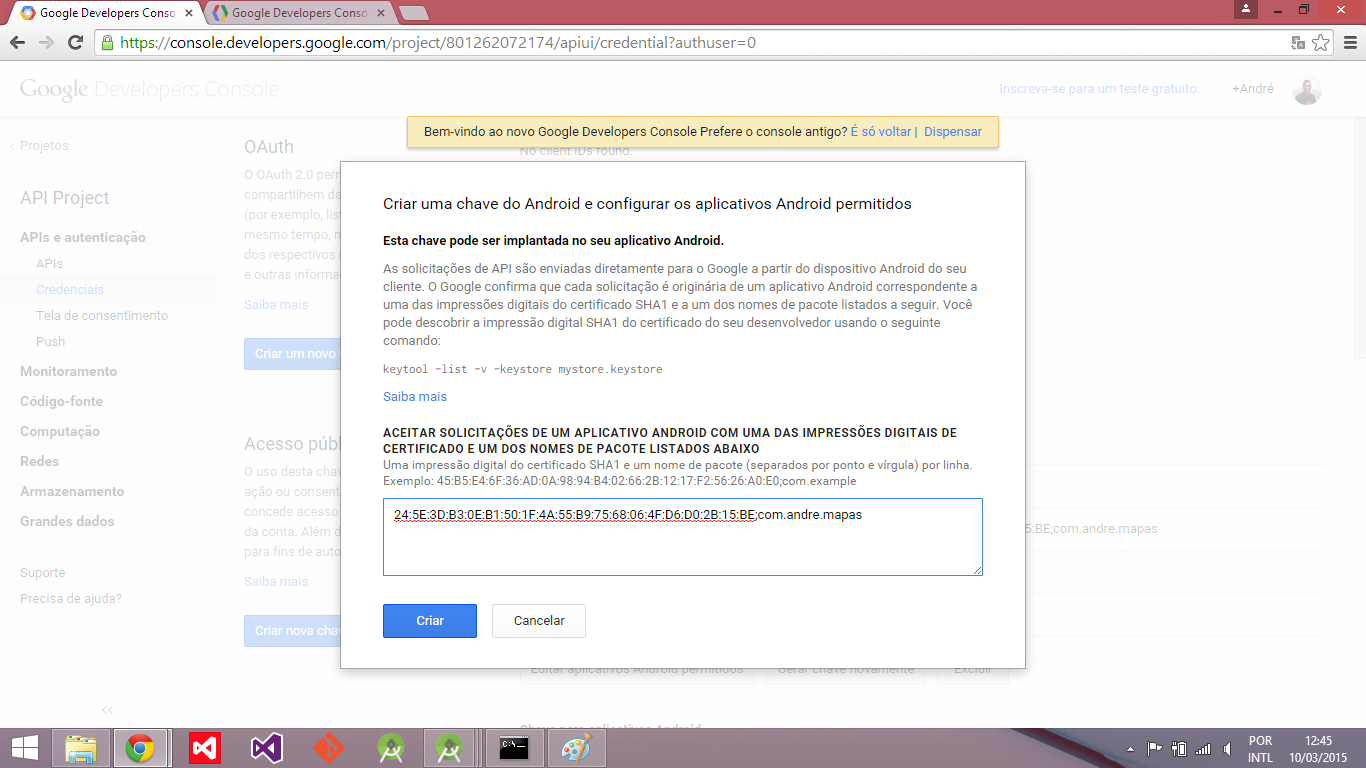
Figura 26: Api Key



Fonte: Print Screen da janela do Prompt de Comando (2015).

Com a chave copiada, basta retornar ao site da api da Google e colar sua chave no campo disponível, utilizando a separação de “ ; “ (ponto e virgula) e o nome do pacote criado e clicar em criar, confome figura 27.

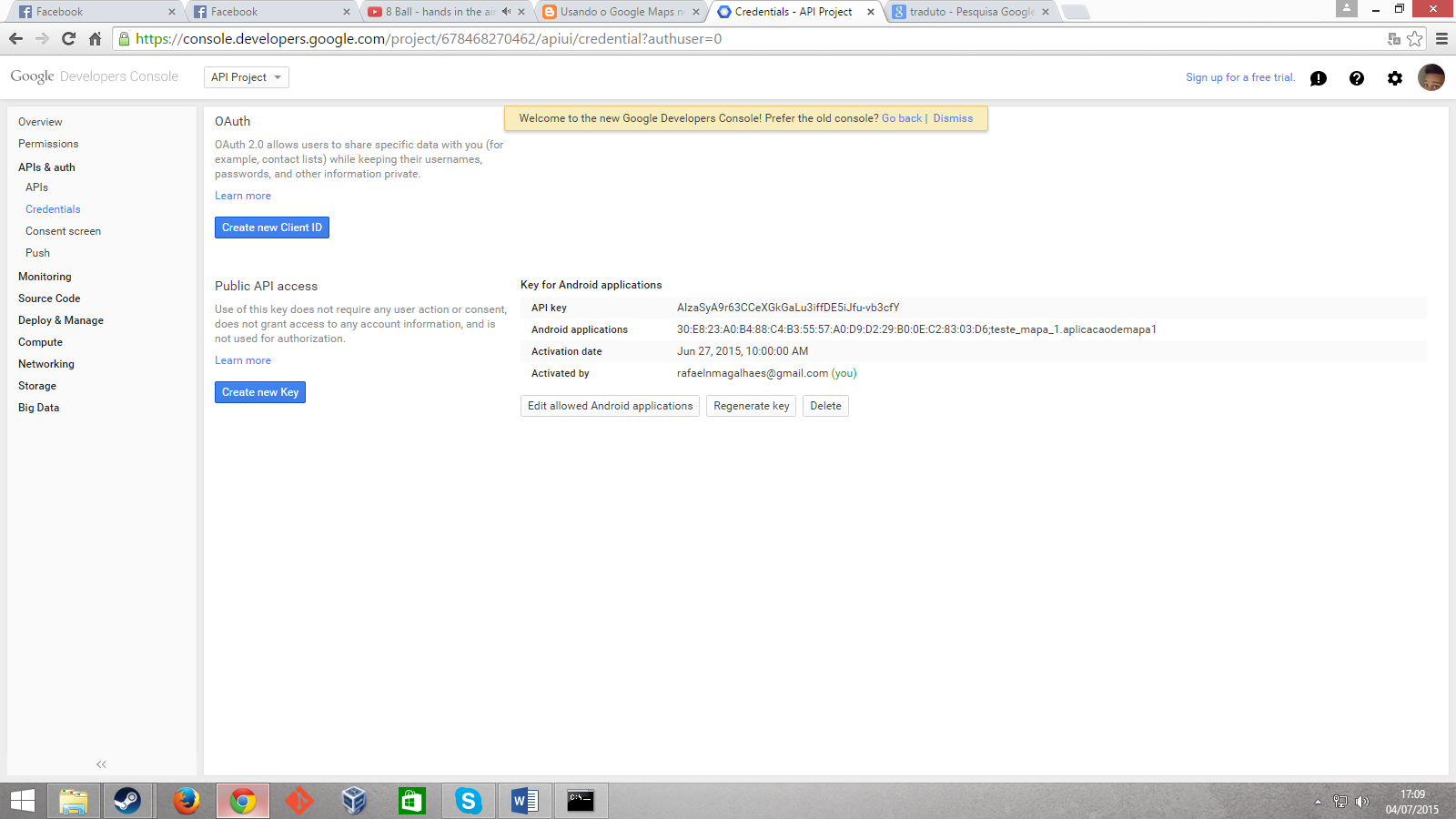
Figura 27: Api Key



Fonte: Print Screen da criação da API Key (2015).

Será criada uma chave da api key, copie a chave para ser inserida na sua aplicação, a tela sera igual a mostrada na figura 28.

Figura 28: Api Key



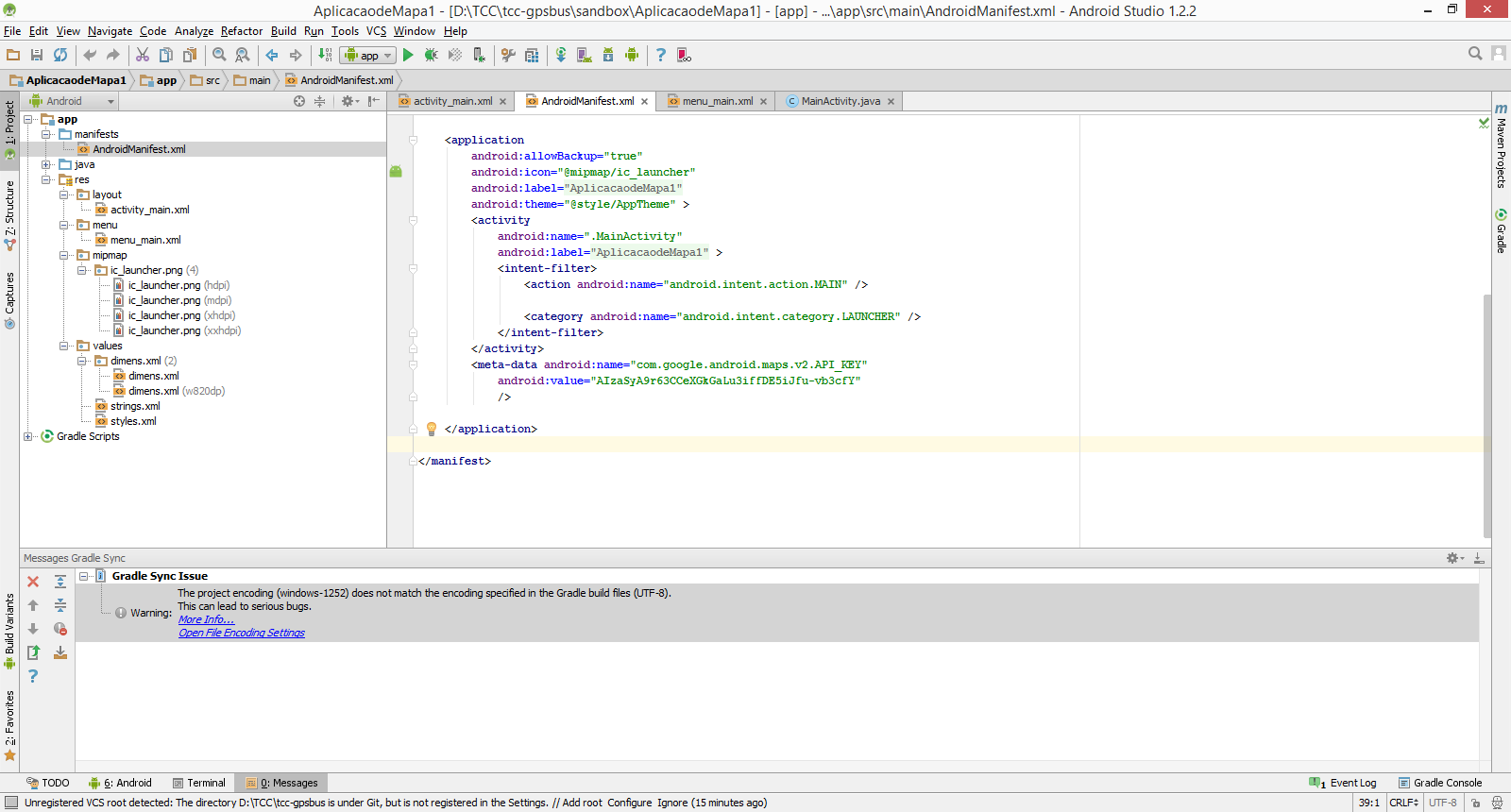
Fonte: Print Screen da criação da API Key (2015).

Retornando ao Android Studio, no projeto, deve-se procurar pela pasta “manifest” e “AndroidManifest.xml”, nesse arquivo será criado uma meta-data para ser inserido a chave da API Key para o funcionamento do mapa. Aberto o arquivo, será inserido a seguinte linha de comando:

<**meta-data android:name="com.google.android.maps.v2.API\_KEY"  
 android:value="AIzaSyA9r63CCeXGkGaLu3iffDE5iJfu-vb3cfY"** />

A imagem 29 ilustra onde sera inserido o código.

Figura 29: Api Key



Fonte: Print Screen da Inserção do código da API Key (2015).

Após esse procedimento, teremos um mapa funcional. Para execução basta executar via Android Studio.

# BIBLIOGRAFIA

Android, D. (2015). *Emulator*. Acesso em 03 de Julho de 2015, disponível em Developers Android: http://developer.android.com/tools/devices/emulator.html#accel-vm

Silveira, F. (02 de Maio de 2010). *Activity - O que é isso?* Acesso em 03 de Julho de 2015, disponível em Felipe Silveira: http://www.felipesilveira.com.br/2010/05/activity-o-que-e-isso/

1. UI – User Interface ou em português Interface do Usuario [↑](#footnote-ref-1)
2. API KEY – Application programming interface key ou em português Chave de interface de programação de aplicação [↑](#footnote-ref-2)