

GOOGLE PLAY GAMES SERVICES



*Simplificando o **DESENVOLVIMENTO** de seu
JOGO ANDROID*

ÍNDICE

Introdução		
Os Conceitos de Jogo		
Achievements		
Regras para pontuação	14	
Ganhando pontos de experiência (XP)	15	
Leaderboards		
Múltiplas leaderboards	16	
Leaderboards públicos e sociais	18	
Real-Time Multiplayer		
Iniciando uma sala	19	
Configuração da sala	20	
Os participantes	22	
Auto-matching	24	Utilizando o Play Games Services
Convites	25	Antes de você começar
Gameplay	25	Configurando as bibliotecas do Google Play Services
Fechamento da sala	26	Iniciando o projeto
Turn-based Multiplayer	26	Fazendo login na conta do Google
Events e Quests	28	Trabalhando com Achievements
Saved Games	29	Criando um Achievement
Imagens de capa	33	Implementando um Achievement
Descrição	38	Trabalhando com Leaderboards
Cota	39	Criando um Leaderboard
Proteção de escrita/leitura	40	Implementando um Leaderboard
Suporte offline	40	
Resolução de conflitos	40	
Limites	40	Plataformas e Game Engines
	41	Plataformas
	42	Game Engines
	43	
		Conclusão
		99



Introdução

Este ebook tem o objetivo de esclarecer e dar o pontapé inicial em como aproveitar os recursos e serviços disponibilizados pelo **Google Play Games** para impulsionar e melhorar seus jogos.

Antes de mais nada, existem dois conceitos que devemos entender: o **Google Play Games** e o **Play Games Services**. No final, eles são “a mesma coisa”, mas é importante conhecer a responsabilidade de cada um separadamente primeiro para depois entender o todo.



O que é o Google Play Games?



O **Google Play Games** é um aplicativo/serviço que o usuário de jogos tem acesso para acompanhar toda sua evolução dentro dos games. O que ele faz é agir como um painel de controle para todas as suas necessidades de jogos no seu dispositivo **Android**.



Pense nisso como o painel do **Xbox**, **PS3** ou mesmo o painel do **Steam**.

Usando esse aplicativo você pode visualizar todos os seus jogos que têm suporte ao **Google Play Games** e também pode ver coisas como conquistas, jogos salvos, fases concluídas e etc. Você também pode ver todos os jogos anteriormente jogados, aqueles que jogou recentemente, e aqueles que tem atualmente instalados. É possível também navegar por títulos mais populares e em destaque, assim como os títulos que tem o apoio oficial do Google para jogar em formato multiplayer.

Um dos pontos mais fortes é você poder visualizar seus amigos no Google+ que também usam o Google Play Games. Isso é ótimo para encontrar amigos que gostam dos mesmos jogos que você para poderem jogar juntos competindo entre si ou se ajudando.



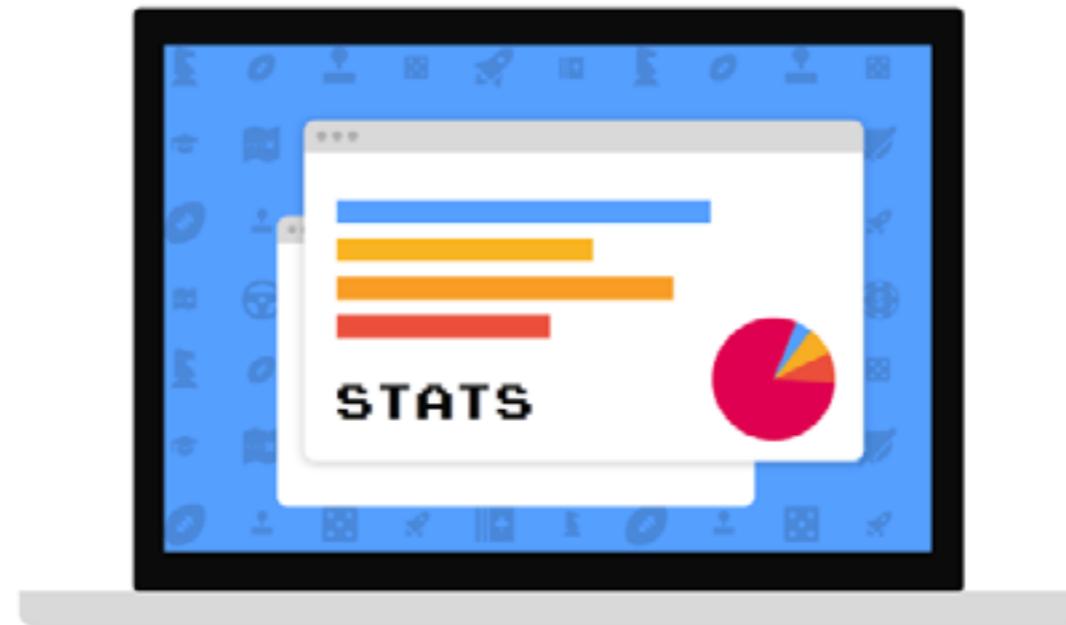
Em resumo, ele é um painel gráfico onde os jogadores podem encontrar tudo o que precisam sobre os jogos em um só local e também para amigos se divertirem juntos.

Para finalizar, o Google Play Games é uma ferramenta muito poderosa tanto para os usuários como para os desenvolvedores de jogos.

Ele dá aos jogadores e desenvolvedores a sua própria interface para monitorar e interagir com seus jogos em um só lugar. A plataforma é grátis e está crescendo e melhorando cada vez mais. Daqui um tempo ela se tornará um aplicativo essencial para qualquer jogador mobile.



O que é o Play Games Services?



O **Play Games Services** é essencial na construção de uma plataforma de jogos da nova geração. Os recursos disponíveis podem fazer seus jogos muito mais sociais, contando com achievements, leaderboards e multiplayer, bem como mais poderoso também. E ainda é possível armazenar jogos salvos e configurações na nuvem.



Você pode utilizar esses recursos tanto no **Android** como no **iOS** ou em qualquer outro dispositivo.

Todos os serviços foram construídos utilizando as melhores tecnologias do Google para mobile e para a nuvem, assim você pode se concentrar no que você realmente é bom como desenvolvedor de jogos: **a criação de grandes experiências de jogos para os usuários.**



Os Conceitos de Jogo

O **Play Games Services** dá suporte a várias necessidades e conceitos no mundo dos jogos. Cada um é tratado como um serviço diferente que pode ser implementado de forma separada.

Você não é obrigado a utilizar todos os serviços, mas é muito importante que você saiba que eles existem para poder agregar em seu jogo em um futuro.

Nesse tópico vamos entender um pouco de cada conceito e como o **Play Games Services** trata eles dentro da plataforma do Google.



Achievements

Esse tipo de mecânica pode ser uma ótima maneira de aumentar o envolvimento dos usuários dentro do seu jogo.

Você pode implementar os achievements em seu jogo para incentivar os jogadores a alcançar alguns recursos que eles não têm acesso normalmente ou qualquer outro desafio que mantenha o usuário a continuar tentando atingir seus objetivos dentro do game.

Os achievements também podem ser uma forma divertida para os jogadores puderem comparar o seu progresso uns com os outros e gerar uma competição.



Como funciona?

Os achievements podem ser atribuídos ao jogador de forma padrão ou incremental. Geralmente, um achievement incremental envolve um jogador fazendo progresso gradual para atingir seus objetivos durante um longo período de tempo.

Conforme o usuário progride para atingir o achievement, você pode guardar o progresso parcial do jogador no **Play Games Services**.

O **Play Games Services** mantém um registro do progresso dentro do jogo, avisa quando o jogador preenche os critérios necessários para desbloquear esse achievement e diz o quanto longe ele está de cumprir esse objetivo.



Os achievements incrementais são cumulativos entre as sessões de jogo e o progresso não pode ser removido ou redefinido dentro do jogo.

Por exemplo, “Ganhar 50 jogos” é considerado como um achievement incremental, mas “Ganhar 3 jogos seguidos” não seria, porque o progresso do jogador seria redefinido quando ele perdesse um jogo.

“Tenha 5.000 fichas de poker” também não seria, pois um jogador pode ganhar e perder as fichas no decorrer dos jogos. Ao criar um achievement incremental, é necessário definir o número total de passos necessários para desbloqueá-lo (este deve ser um número entre 2 e 10.000).



À medida que o usuário faz progressos para desbloquear o achievement, você deve informar o número de passos realizados pelo usuário ao Play Games Services.

Uma vez que o número total de passos atinge o valor de desbloqueio, o achievement é desbloqueado.

Um jogo deve ter pelo menos cinco achievements antes de ser publicado. Você pode testar com menos de cinco achievements, mas você precisa de pelo menos cinco criados antes de publicar o seu jogo na Google Play Store.



Regras para pontuação

Todos os achievements têm um valor de pontuação associados a eles. A pontuação do jogador deve ser um múltiplo de 5 e um jogo não pode ter um total de mais de 1000 pontos juntando todas as seus achievements. Além disso, nenhum achievement pode ter mais de 200 pontos.



Ganhando pontos de experiência (XP)

Os jogadores podem conseguir níveis no seu perfil quando ganham achievements no Play Games Services.

Para cada ponto associado com um achievement, o jogador ganha 100 pontos de experiência (XP) quando ele atinge um achievement. Veja a conta abaixo:

$$\text{XP de um achievement} = 100 * (\text{valor de pontos do achievement})$$

O **Play Games Services** mantém o controle do XP ganho por cada jogador e envia uma notificação para o aplicativo da Google Play Games quando o jogador tem pontos suficientes para “subir de nível”. Os jogadores podem ver o seu nível e o histórico de XP a partir de seu perfil no Google Play Games.



Leaderboards

Os leaderboards podem ser uma maneira divertida de gerar concorrência entre os seus jogadores, tanto para os seus fãs mais hardcore quanto para os jogadores mais casuais.

Como funciona?

Quando você cria um leaderboard, o Play Games Services gerencia a maioria dos dados deste leaderboard para você. No final de um jogo ou em algum momento determinado por você, o jogo apresenta a pontuação do jogador no leaderboard que você criou.



O Play Games Services verifica se o resultado atual é melhor do que o corrente no leaderboard do jogador. Se for, o serviço atualiza as tabelas de classificação correspondentes com a nova pontuação.

O Play Games Services envia um relatório de pontuação para o jogador dizendo se a pontuação adquirida é a melhor dentro das estatísticas diárias, semanais ou de todos os tempos. Se não for, o serviço vai dizer ao jogador o seu desempenho baseado nas estatísticas diárias, semanais ou de todos os tempos.

Para recuperar os resultados de um jogador, você deve solicitar um prazo (diária, semanal, ou de todos os tempos) e especificar se o usuário quer ver um leaderboard baseado nas redes sociais ou público.



Múltiplas leaderboards

Os jogos podem ter várias leaderboards até um máximo de 70. Por exemplo, um jogo multi-nível pode oferecer um leaderboard diferente para cada nível e um jogo de corrida pode ter um ranking separado para cada pista.



Leaderboards públicos e sociais

Existem dois tipos diferentes de visualização do leaderboard do jogador:

A **classificação social** é uma leaderboard composta de pessoas nos círculos do usuário (ou, mais precisamente, os membros dos círculos que o usuário escolheu para compartilhar com sua aplicação) que decidiram compartilhar sua evolução dentro dos jogos com outros usuários.

Já o **leaderboard público** é um leaderboard formado por jogadores que optaram por compartilhar a sua evolução nos jogos de forma pública. Se o jogador não tiver escolhido compartilhar a sua evolução no jogo publicamente, ele não vai aparecer neste ranking.



Real-Time Multiplayer

Seu jogo pode usar a **API multiplayer** do Play Games Services para conectar vários jogadores juntos em uma única sessão de jogo e de transferência entre jogadores conectados.

Usar a **API real-time multiplayer** pode ajudar a simplificar o seu esforço de desenvolvimento do jogo, porque a API lida com várias tarefas para você.

- Ela gerencia as conexões de rede para criar e manter um ambiente multiplayer em tempo real, ou seja, permite a comunicação de rede entre vários jogadores na mesma sessão de jogo e permite aos jogadores enviar dados diretamente um para o outro
- Fornece uma interface de usuário para convidar outros jogadores a entrar em uma sala



- Guarda todas as informações dos participantes e das salas multiplayer durante o ciclo de vida do jogo
- Envia convites das salas multiplayer e atualizações para os jogadores, notificando todos os dispositivos em que o jogador está logado

Como funciona?

Antes de implementar o seu jogo usando a API do real-time multiplayer, você deve se familiarizar com os conceitos a seguir, que estão relacionados com o ciclo de vida de um jogo multiplayer em tempo real.



Iniciando uma sala

Internamente, uma sala é uma configuração de rede utilizando o protocolo **Peer-to-Peer** entre os participantes, onde os clientes podem se comunicar diretamente uns com os outros.

Antes de uma sessão de jogo multiplayer ser iniciada em um dispositivo, o usuário do dispositivo deve estar conectado ao seu jogo.

O jogador local (ou seja, o usuário que está conectado ao dispositivo onde o jogo está em execução) pode, então, iniciar uma sessão de jogo multiplayer, convidando amigos para participar do jogo ou pedir para ser encontrado aleatoriamente.

A API fornece uma interface gráfica que permite aos jogadores convidar seus amigos ou selecionar um número máximo de adversários que podem encontrar e participar da sessão.



Isso simplifica sua codificação na hora de desenvolver uma interface gráfica, mas você também pode optar por implementar isso de forma manual.

Com base nos detalhes de seleção de jogador e nas configurações da sala, o Play Games Services tentará criar uma sessão de jogo multiplayer em tempo real.

Se a sala for criada com sucesso, o Play Games Services notifica o jogo e o jogador local é automaticamente colocado como participante da sala.



Configuração da sala

Você deve especificar o número de jogadores que você deseja permitir na sala. Atualmente, o Play Games Services suporta um máximo de oito jogadores em um jogo multiplayer (incluindo o jogador que está iniciando a partida).

Opcionalmente, você pode querer garantir que apenas os jogadores que estão interessados em um tipo específico de variante de jogo encontrem sua sala. Por exemplo, em um jogo de corrida, você pode encontrar jogadores que só querem jogar um mapa de corrida ou nível de dificuldade específica.

As variantes podem ser utilizados para os jogadores que estão interessados em estilos de jogo diferentes encontrarem sua sala. Se existem diferentes versões do seu aplicativo, você também pode usar variantes para garantir que apenas os jogadores que estão em versões compatíveis consigam se encontrar.



Os participantes

Quando os jogadores iniciam um jogo multiplayer, eles podem optar por convidar pessoas específicas ou deixar o Play Games Services selecionar automaticamente outros participantes aleatoriamente através do auto-matching.

Eles também podem solicitar uma mistura dos dois (por exemplo, um jogador específico de seus círculos, e dois jogadores via **auto-matching**).

Auto-matching

Um participante encontrado via **auto-matching** não tem que ser um contato dos círculos do jogador local ou qualquer outra conexão. O auto-matching simplesmente procura por outros participantes que também estão iniciando um jogo naquele momento e pedindo para serem encontrados.



Convites

Um usuário de dispositivo móvel que recebe um convite verá uma notificação no dispositivo onde está logado. Os convites são enviados pelo Play Games Services através de mensagens via **Google Cloud** para dispositivos Android, e através do serviço de mensagens da Apple (APNS) para dispositivos iOS .

Se o jogador não tiver o aplicativo instalado em um dispositivo Android, eles serão solicitados a instalar o aplicativo a partir da Google Play Store. Nesse caso, o convite fica aguardando e o jogador pode aceitá-lo depois de instalar o jogo.

Gameplay

Uma vez que o número necessário de participantes para uma sala tenha sido atingido, a sala é considerada como “cheia” e o jogo pode começar.



Em determinados cenários avançados, o jogo pode permitir que os participantes conectados iniciem o jogo antes mesmo de todos os convites pendentes serem aceitos.

Se o seu jogo suporta este modo de jogo, certifique-se de lidar com quaisquer participantes que entrarem na sala após o jogo estar em andamento.

Veja o seguinte exemplo:

Em um jogo de corrida de 3 jogadores, a sua sessão pode começar com dois jogadores. Durante a corrida, se um terceiro jogador entra na sala, o game pode deixar o participante recém chegado apenas observando corrida atual como um espectador, mas não podendo jogar como um piloto. Após a corrida acabar, o jogo pode permitir que os três jogadores participem como pilotos na próxima rodada.



Fechamento da sala

É de responsabilidade do seu game avisar os servidores do Play Games Services quando um jogador que está participando da sala sair. Seu jogo deve também lidar com o cenário em que todos os participantes exceto o jogador local deixem a sala. Quando isso acontece, o jogo deve desconectar o jogador local da sala imediatamente.

A sala é considerada “fechada” quando todos os seus participantes saem dela. Neste ponto, o jogo deve desligar qualquer jogo em andamento, e certificar-se de salvar os dados de forma adequada.



Turn-based Multiplayer

Em um jogo multiplayer baseado em turnos, um único estado compartilhado é passado entre vários jogadores, e apenas um jogador tem permissão para modificar o estado compartilhado de cada vez.

Os jogadores se revezam de forma assíncrona de acordo com uma ordem de jogo determinado pelas regras do jogo.

Seu jogo pode usar a API turn-based multiplayer fornecida pelo Play Games Services para gerenciar as seguintes tarefas:

- Convidar jogadores para participar de uma partida multiplayer baseado em turnos. Procure por jogadores de forma aleatória para serem adicionados automaticamente no seu jogo. É permitido colocar até oito participantes em uma partida



- Guardar informações sobre o jogadores e a partida nos servidores do Google e compartilhar os dados atualizados de forma assíncrona com todos os participantes ao longo do ciclo de vida do jogo
- Enviar convites e notificações da partida aos jogadores

Como funciona?

Um jogo baseado em turnos é uma sessão de jogo com vários participantes que se revezam de forma consecutiva fazerem suas jogadas durante a partida. As partidas devem ser iniciada por um jogador logado no Play Games Services.



Seu jogo pode usar a API turn-based multiplayer para se juntar até oito jogadores em uma mesma partida, incluindo o jogador que iniciou e quaisquer jogadores encontrados automaticamente.

As partidas acontecem de forma assíncrona e os participantes não precisam estar conectados simultaneamente ao Play Games Services para jogar.

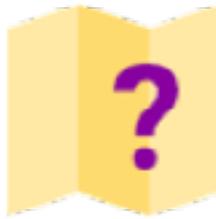
Existem três coisas básicas que formam uma partida baseada em turnos:

- **Participantes:** um usuário pode se tornar um participante em um jogo baseado em turnos, iniciando uma partida, aderirindo a um jogo ao aceitar um convite, ou utilizar o auto-matching



- **Game data:** conforme um jogo avança, o jogador pode modificar e armazenar os dados nos servidores do Google. Os outros participantes podem, em seguida, recuperar e atualizar esses dados na sua vez do turno
- **Match state:** um jogo pode ter um dos seguintes estados: active, auto-matching, complete, canceled e expired, dependendo das ações dos participantes durante a partida. O estado de um jogo é gerido pelo Play Games Services

Seu jogo pode verificar o estado do jogo para determinar se um jogo pode continuar, se os jogadores podem entrar na partida e se o jogo terminou.



Events e Quests

O Play Games Services permite recolher dados cumulativos gerados por seus jogadores durante o jogo e armazená-los nos servidores do Google para análise do jogo.

Você pode definir com flexibilidade os dados do jogador que seu jogo deve recolher:

- Os jogadores usam um determinado item
- Os jogadores atingem um certo nível
- Os jogadores executam alguma ação do jogo em específico

Você pode usar os dados de eventos como feedback sobre como melhorar o seu jogo. Por exemplo, você pode ajustar a dificuldade de certos níveis em seu jogo que os jogadores estão encontrando muita dificuldade para completar.



O Play Games Services complementa o serviço de eventos com as quests, permitindo-lhe introduzir novos desafios com prazos que são baseados em dados de eventos. As quests lhe permitem envolver os jogadores e incentivá-los com alguma recompensa no jogo ou se beneficiar se eles tiverem sucesso, sem ter que publicar seu jogo inteiro novamente.

Como funciona?

As APIs de eventos fornecem uma maneira de definir e coletar métricas de jogabilidade interessantes e fazer o upload dessas métricas o Play Games Services.

Por exemplo, veja como seria um evento que poderia ser enviado para o serviço dizendo que o jogador ganhou moedas de ouro por ter matado os zumbis.



Evento:

- Nome: Zumbis Mortos
- Descrição: Número de vezes que um jogador matou um zumbi
- Tipo: Ganhou moedas de ouro

Ou então, o jogador pode ter perdido moedas de ouro por ter sido atingido por um zumbi.

Evento:

- Nome: Ataque Zumbi
- Descrição: Número de vezes que um jogador foi atingido por um zumbi
- Tipo: Perdeu moedas de ouro



Esses dois casos exemplificam os dois tipos de eventos que podem ser usados, o Premium currency source, onde um jogador ganha pontos/moedas/ouro por fazer determinada ação ou Premium currency sink, quando o jogador gasta seus recursos ou perde por algum motivo. As APIs de quests permitem que o desenvolvedor crie desafios dentro do jogo para os jogadores para tentarem completar dentro de um período de tempo pré-definido. Vamos ver alguns exemplos de quests que podemos utilizar no serviço.

Quest:

- Nome: Coletar 50 moedas
- Descrição: Os zumbis estão tomado a cidade neste fim de semana. Ajude matando todos os zumbis
- Critérios de conclusão: Mate 100 zumbis



Ou então:

Quest:

- Nome: Comer 100 panquecas
- Descrição: Nessa semana temos a competição “Panqueca Comilão”, vença o torneio comendo panquecas
- Critérios de conclusão: Coma 100 panquecas para vencer o torneio



Saved Games

O serviço de **Saved Games** é uma maneira fácil salvar a progressão dos seus jogadores no jogo nos servidores do Google. Seu jogo pode recuperar os dados salvos para permitir aos jogadores retornarem de onde pararam a partir de qualquer dispositivo.

Você pode sincronizar os dados de um jogador em vários dispositivos diferentes. Por exemplo, se você tem um jogo que roda em Android, você pode usar o serviço para permitir que um jogador comece um jogo em seu telefone Android e depois continue a jogar em um tablet sem perder seu progresso.

Este serviço também pode ser usado para garantir que um jogador continue a partir de onde ele parou, mesmo se o seu dispositivo for perdido, destruído, ou trocado por um modelo mais novo.



Como funciona?

O serviço Saved Games é dividido em duas partes:

1. Sistema binário não estruturado: esses dados podem representar o que você quiser, e seu jogo é responsável por analisar e escrever esses dados.
2. Metadados estruturados: são propriedades adicionais associados com os dados binários que permitem que o Play Games Services para apresentar visualmente os Saved Games para os usuários.

Imagens de capa

O serviço fornece ao usuário uma experiência visual aos dados salvos. Você pode associar imagens para representar os jogos salvos. Se você estiver usando a interface padrão para mostrar os jogos salvos, ele irá exibir essas imagens. As imagens também podem aparecerem no Google Play Games.



Descrição

Você pode fornecer uma descrição breve do conteúdo de um jogo salvo. Esta descrição é exibida diretamente para os jogadores e deve resumir o estado em que o jogo salvo representa; por exemplo, "Salvando a princesa Lela".

Cota

Os desenvolvedores não são cobrados pelos dados de jogos salvos que estão armazenado na nuvem. Em vez disso, esses dados são consumidos diretamente da conta do Google Drive do jogador - você nunca tem que se preocupar com isso.

Proteção de escrita/leitura

Todos os jogos guardados são armazenados no Google Drive em uma pasta de dados dos seus jogadores.



Esta pasta só pode ser lida e escrita por seu jogo - não pode ser vista ou modificada por outros jogos, para que haja uma proteção contra a corrupção de dados. Além disso, os jogos salvos são isolados para que eles não possam ser modificados pelos próprios jogadores.

Suporte offline

O seu jogo ainda pode ler e escrever em um jogo salvo mesmo quando o dispositivo do jogador está sem conexão com a internet, mas não será capaz de sincronizar os dados com o Play Games Services até que a conectividade de rede seja estabelecida. Uma vez restabelecida, o Play Games Services, de forma assíncrona atualiza os dados dos jogos salvos nos servidores do Google.



Resolução de conflitos

Ao usar o serviço Saved Games, o jogo pode encontrar alguns conflitos ao tentar salvar os dados. Estes conflitos podem ocorrer quando um usuário estiver executando mais de uma instância de sua aplicação em diferentes dispositivos ou computadores. Sua aplicação deve ser capaz de resolver esses conflitos de uma forma a proporcionar a melhor experiência para o usuário.

Normalmente, os conflitos de dados ocorrem quando uma instância do seu aplicativo é incapaz de conectar no serviço Saved Games durante uma tentativa de carregar os dados ou salvá-los.

Em geral, a melhor maneira de evitar conflitos de dados é sempre carregar os dados mais recentes do serviço quando o aplicativo é iniciado e salvar os dados usando o serviço com uma frequência razoável.



No entanto, isso nem sempre é possível, sua aplicação deve fazer todos os esforços para evitar os conflitos de tal modo que os dados dos usuários sejam preservados e que eles tenham uma boa experiência.

Limites

O Play Games Services tem um limite do tamanho de dados guardados e da imagem que representa esses dados sendo 3 MB e 800 KB, respectivamente.



Como configurar o Play Games Services

Depois de aprender vários conceitos e entender o que o Play Games Services pode fazer, chegou a hora de começar a configurar o serviço para podermos usá-lo em nossos jogos.

A configuração é feita no Google Play Developer Console e serve para seus jogos Android, iOS, C++ ou jogos baseados na web. O Google Play Developer Console fornece um local centralizado para você gerenciar os serviços de jogos e configurar os metadados para autorizar e autenticar o seu jogo.



Para adicionar o seu jogo no Google Play Developer Console, siga estes passos abaixo.

#1 Passo - Faça login no Google Play Developer Console

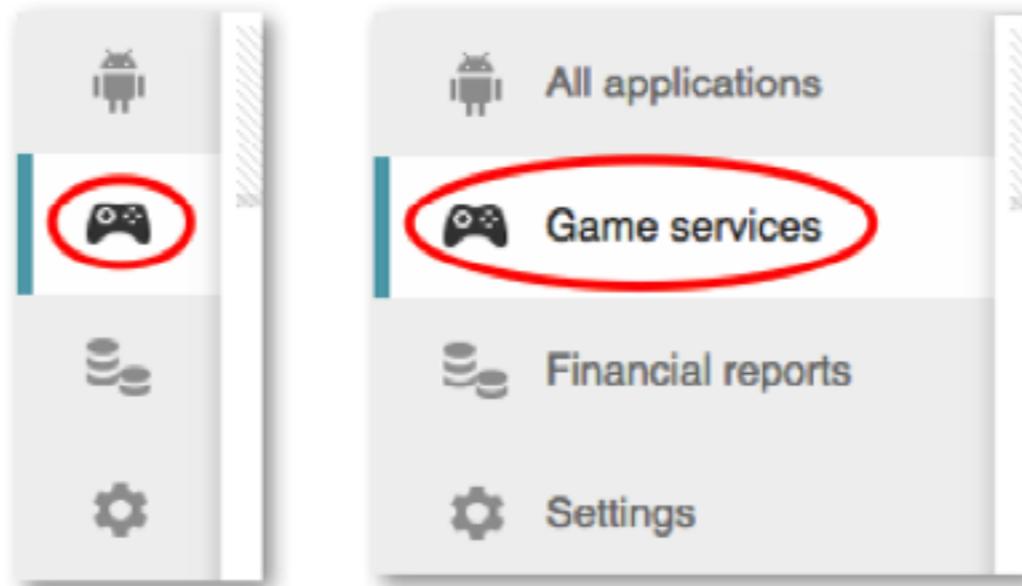
Para fazer o login, acesse o [Google Play Developer Console](#). Se você ainda não tem uma conta, faça o cadastro clicando em “Criar uma conta”.



#2 Passo - Adicionar o seu jogo ao Google Play Developer Console

Para adicionar seu jogo, siga estes passos:

1. Abra o menu **Game Services**, selecione a guia do lado esquerdo, em seguida, clique no botão **Set up Google Play game services** ou **Add new game**.





2. Existem duas formas de adicionar um jogo.

Se você está criando um jogo do zero, ou se você nunca usou uma das APIs do Google anteriormente, escolha a aba **I don't use any Google APIs in my game yet**.

Digite o **nome** do seu jogo e escolha uma categoria e, em seguida, clique no botão **Continue**.

SET UP GOOGLE PLAY GAME SERVICES FOR AN APP

Do you already use Google APIs in your app?

I don't use any Google APIs in my game yet I already use Google APIs in my game

What is the name of your game?
AndroidPro Game 15 of 30 characters
This is the name that will be displayed to users in Google Play game services.

What kind of game is it?
Educational ▾
The category helps users browse interesting games.

Google Play game services use the following APIs: Google+ API, Google Play Game Services and Google Play Game Management
We will automatically create a project for your game on the [Google Developers Console](#) and enable the necessary APIs for you.

Continue **Cancel**



SET UP GOOGLE PLAY GAME SERVICES FOR AN APP

Do you already use Google APIs in your app?

I don't use any Google APIs in my game yet

I already use Google APIs in my game

If your game already uses Google APIs, you have created a project for the game on the [Google Developers Console](#) before.

Choose an API console project to link

api-androidPro

I can't see my project in the list

What kind of game is it?

Educational

The category helps users browse interesting games.

Continue

Cancel

Se este é um jogo onde você já configurou um ou mais APIs do Google, selecione a aba **I already use Google APIs in my game.**

Você verá uma lista de projetos do Google Developers Console.

Selecione seu projeto nesta lista, uma categoria e, em seguida, clique no botão **Continue**.



3. Em **Game Detail**, adicione a descrição, categoria e recursos gráficos do seu jogo. Apenas o nome de exibição é necessário para o teste.

AndroidPro Game
776323962148

Draft ▾

Quests

Game details

Linked apps

Events

Achievements

Leaderboards

Testing

Publishing

GAME DETAILS

Saved

English (United States) – en-US Add translations

Display name *

English (United States) – en-US

AndroidPro Game
15 of 30 characters

Description

English (United States) – en-US

0 of 4000 characters

Category

Educational

Saved Games

ON OFF

GRAPHIC ASSETS

Please add all the graphic assets described below or [use graphic assets](#) from one of your Android apps.
Note: Games with blank or single color graphic assets will not be featured on the Google Play Games app.

High-res icon
512 x 512

Feature Graphic
1024 W x 500 H



Os outros campos devem ser preenchidos antes de publicar o seu jogo. O nome para exibição e uma descrição para o seu jogo deve ser genérico o suficiente para aplicar em todas as versões do seu game que compartilham os mesmos serviços do Play Games Services.

4. Clique em **Save** para criar uma nova entrada para o seu jogo no Google Play Developer Console.

#3 Passo - Gerar um ID de cliente OAuth 2.0

Seu jogo deve ter um ID de cliente **OAuth 2.0**, para ser autenticado e autorizado a utilizar os serviços do Play Games Services.

Para configurar a associação entre um ID do cliente e seu jogo, use o Google Play Developer Console para gerar o ID do cliente e vinculá-lo para o seu jogo.

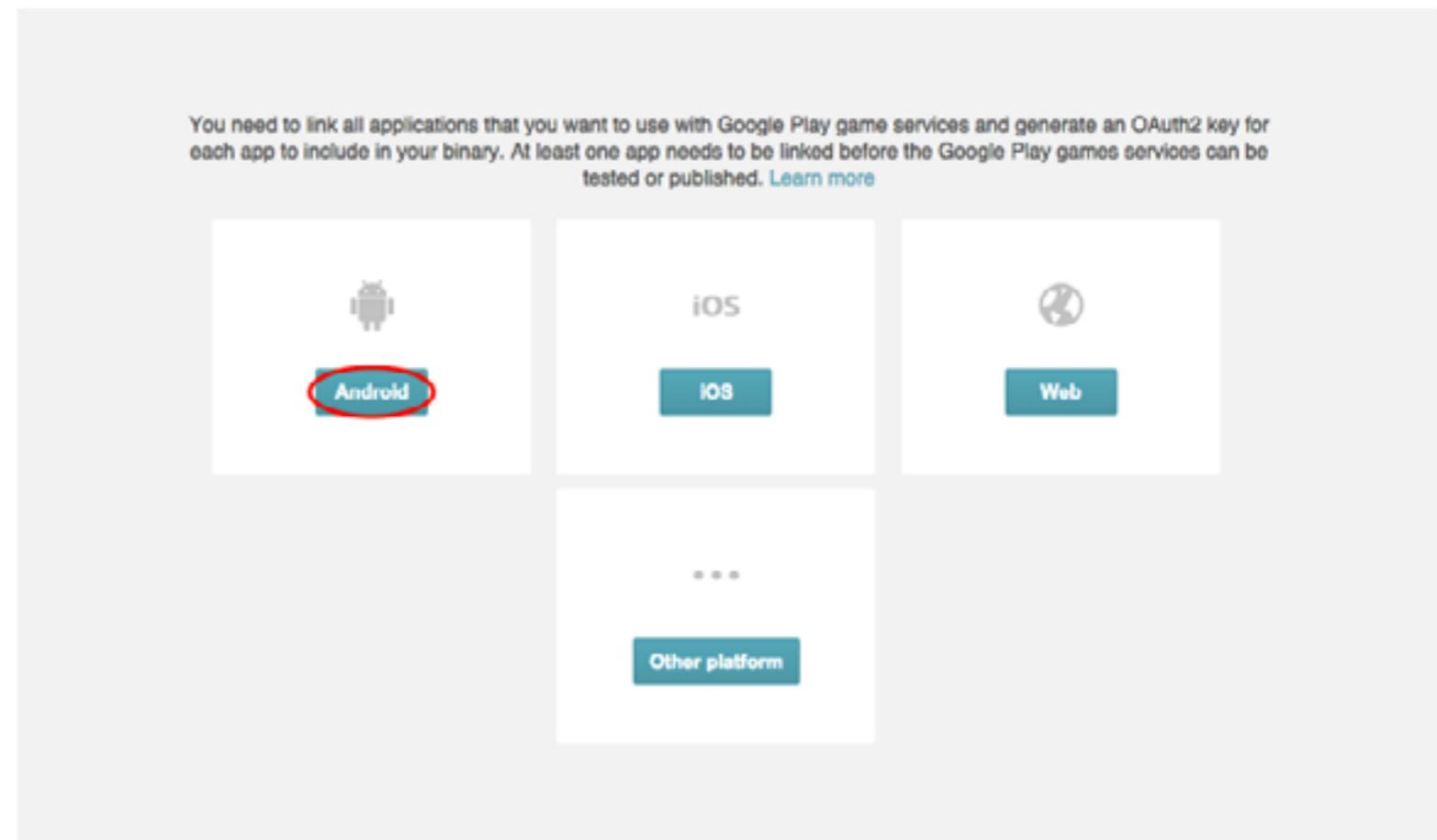


Criar uma Linked App

Para ligar o seu jogo a um projeto do Google Play Developer Console, abra a página **Linked Apps** e siga as instruções específicas para a sua plataforma.

LINKED APPS

1. Clique no botão Android.
2. O nome do aplicativo vai ser mostrado aos seus jogadores. Por isso, escolha um nome que se aproxime do nome real do seu jogo Android.
3. Adicione o pacote do aplicativo.





LINK AN ANDROID APP Save and continue

STEP 1: ENTER THE APP DETAILS

English (United States) – en-US

Name of the app
English (United States) – en-US

AndroidPro Game 15 of 30 characters

Package name
br.com.androidpro.alomundo

MULTIPLAYER SETTINGS

Turn-based multiplayer ON OFF

Real-time multiplayer ON OFF

ANTI-PIRACY

Enable anti-piracy ON OFF

4. Para usar os serviços de real-time ou turn-based multiplayer, habilite as configurações de multiplayer.

5. Escolha se quer ou não ativar a **Anti-Piracy** (anti-pirataria).

6. Clique em **Save and continue**. Em seguida, siga os passos da próxima seção.



Criar um ID de cliente

1. Agora você precisa autorizar seu aplicativo. Clique em **Authorize your app now** para iniciar o processo de criação de um ID de cliente OAuth 2.0

The screenshot shows a web page for 'TYPE-A-NUMBER CHALLENGE'. At the top left is a back arrow icon. Next to it is an Android icon followed by the text 'TYPE-A-NUMBER CHALLENGE'. Below this is a section titled 'STEP 2: AUTHORIZE YOUR APP'. A descriptive text states: 'For your app to work with Google Play games services you need to include an OAuth2 Client ID and Secret in your binaries.' At the bottom of this section is a teal button with white text that reads 'Authorize your app now'.



2. Caso você tenha escolhido um aplicativo que já esteja no Google Play Developer Console e o campo **fingerprint** (certificado digital) já vier preenchido, apenas clique em Confirm, caso contrário, vamos precisar gerar um novo fingerprint (certificado digital).

CREATE ANDROID OAUTH CLIENT

API requests are sent directly to Google from your clients' Android devices. Google verifies that each request originates from an Android application that matches the package name and SHA1 signing certificate fingerprint name listed below.

Package name: br.com.androidpro.alomundo

**Signing certificate fingerprint
(SHA1):** 6F:03:1C:EF:C4:A6:3A:CB:1A:BC:07:89:43:8E:B8:C5:44:E4:B8:25

Confirm

Cancel



Gerar um fingerprint (certificado digital)

Abra um terminal e execute a ferramenta **Keytool** para obter o fingerprint **SHA1** do certificado. Você deve gerar tanto para a versão de produção quanto de desenvolvimento.

Para obter o fingerprint de produção:

```
keytool -exportcert \
-alias <nome-da-sua-chave> \
-keystore <caminho-para-sua-keystore> \
-list -v
```



Para obter o fingerprint de desenvolvimento (debug):

```
keytool -exportcert \
-alias androiddebugkey \
-keystore <caminho-para-keystore-debug> \
-list -v
```

O keytool solicita que você digite uma senha para o armazenamento das chaves. A senha padrão para o armazenamento de chaves de debug é android. O keytool em seguida, imprime o fingerprint no terminal. Por exemplo:

Certificate fingerprint: SHA1: DA:39:A3:EE:5E:6B:4B:0D:32:55:BF:EF:95:60:18:90:A-F:D8:07:09



Cole o fingerprint gerado no campo **Signing certificate fingerprint (SHA1)**. Clique no botão **Confirm**, em seguida, você verá seu ID de cliente.

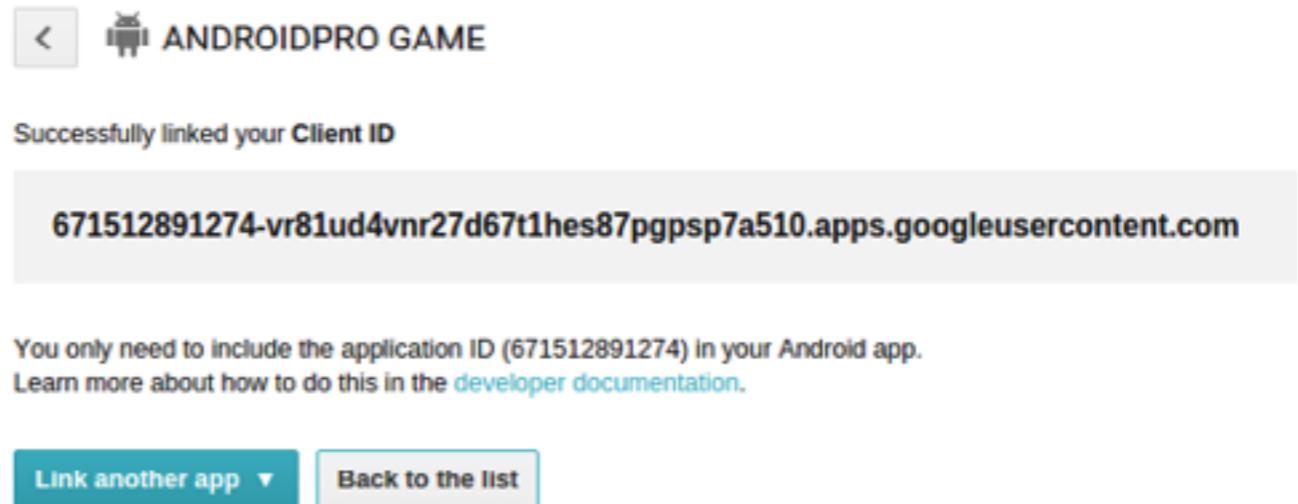
Verificando as credenciais para autenticação e autorização

Dependendo da plataforma que está você esta desenvolvendo, talvez seja necessário localizar e registrar as seguintes informações de credenciais.



ID do Cliente

Depois de clicar no botão **Confirm**, você vai ver o seu novo ID de cliente para esta aplicação. Anote o ID do cliente, você vai precisar dessa informação mais tarde.



Você também pode encontrar essa informação acessando a página de **Linked Apps** e clicar no item com o ID do cliente para sua plataforma.

LINKED APPS		Link another app ▾	or	Continue to next step
NAME		DETAILS	OAUTH 2.0 CLIENT	STATUS
	AndroidPro Game	br.com.androidpro.alomundo	✓	Ready to publish



O ID do cliente fica no final da página.

Name of the app English (United States) – en-US	AndroidPro Game	15 of 30 characters
Package name	br.com.androidpro.alomundo	
MULTIPLAYER SETTINGS		
Turn-based multiplayer	ON	OFF
Real-time multiplayer	ON	OFF
ANTI-PIRACY		
Enable anti-piracy	ON	OFF
AUTHORIZATION		
Application ID	671512891274	
OAuth2 Client ID	671512891274-vr81ud4vnr27d67t1hes87pgpsp7a510.apps.googleusercontent.com	



ID do aplicativo

Você também vai precisar saber o seu ID do aplicativo. Você pode encontrar isso olhando para o número de 12-13 dígitos ao lado do nome de exibição do seu jogo na parte superior da página.

NAME	DETAILS
AndroidPro Game	br.com.androidpro.alomundo



Utilizando o Play Games Services

Chegou a hora de colocarmos a mão na massa e vermos como funciona na prática os recursos dessa plataforma de jogos do Google.

A SDK do Play Games Services permite integrar facilmente os recursos e conceitos de jogos que vimos até agora aqui, tais como achievements, leaderboards, saved games e as duas formas de jogos multiplayer, real-time e turn-based.

Sendo esse um ebook introdutório, vamos mostrar um exemplo utilizando a plataforma Android.

Nessa seção vamos criar um aplicativo de exemplo utilizando dois dos conceitos de jogos que vimos dentro do Play Games Services, os achievements e leaderboards.



No final deste tutorial você vai ter um jogo de exemplo totalmente funcional e integrado com os dois serviços citados.

Você pode baixar o código fonte do jogo clicando [neste link](#).

Antes de você começar

1. Configure o seu ambiente de desenvolvimento Android.
2. Configure um emulador ou dispositivo físico rodando o Android 2.3 ou superior para os testes.
3. É recomendável você testar em um dispositivo físico Android. No entanto, se você não tem um dispositivo físico, você pode testar no emulador Android. Para fazer isso, baixe a imagem do emulador que inclui o Google APIs.



Configurando as bibliotecas do Google Play Services

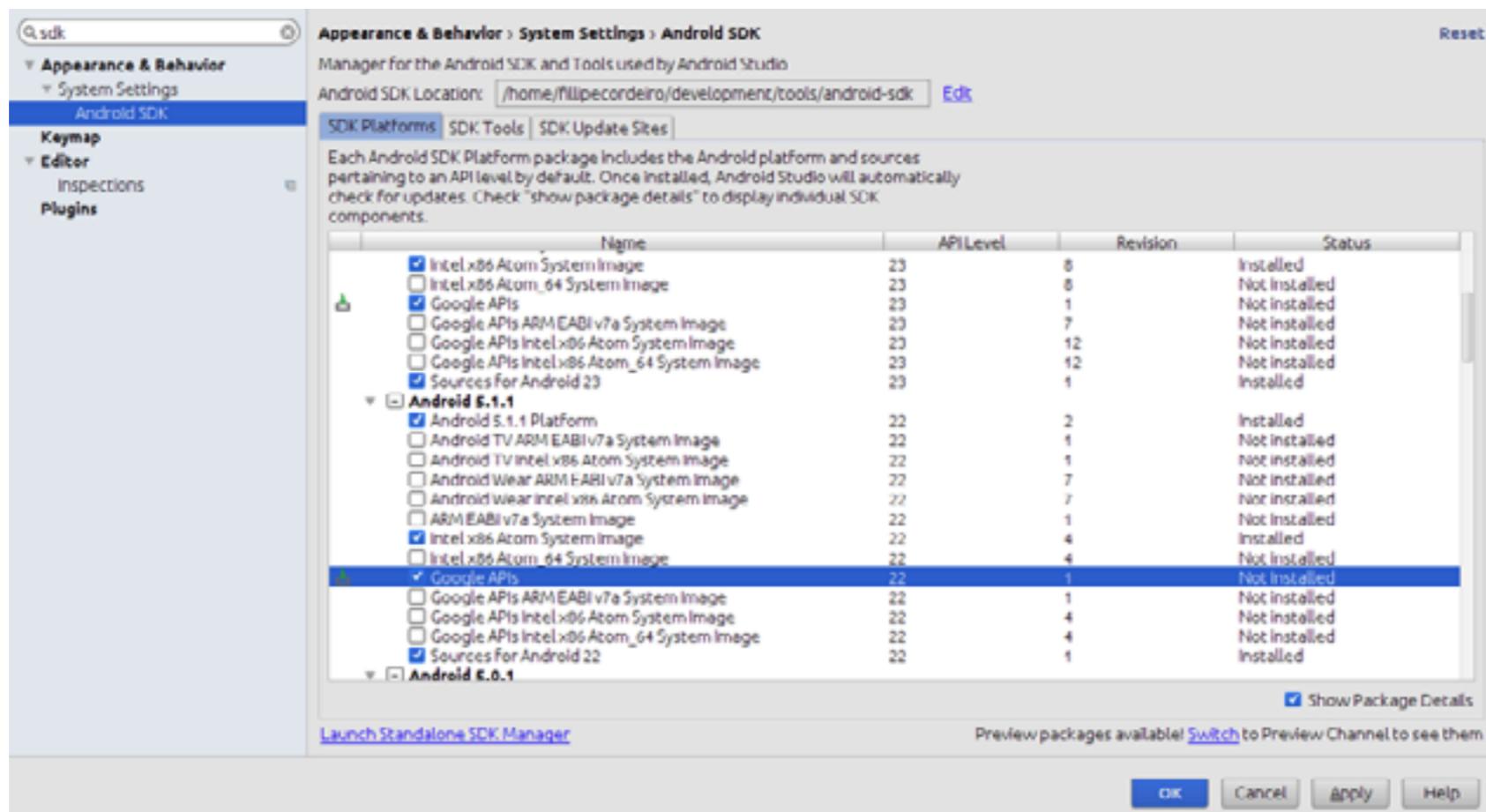
Para começar a utilizar qualquer serviço do Play Games Services no seu jogo ou aplicativo, precisamos fazer a instalação das bibliotecas do Google Play Services utilizando o Android SDK Manager.

Essas bibliotecas servem tanto para os serviços de jogos como para outros como o Maps, Drive e Google+.



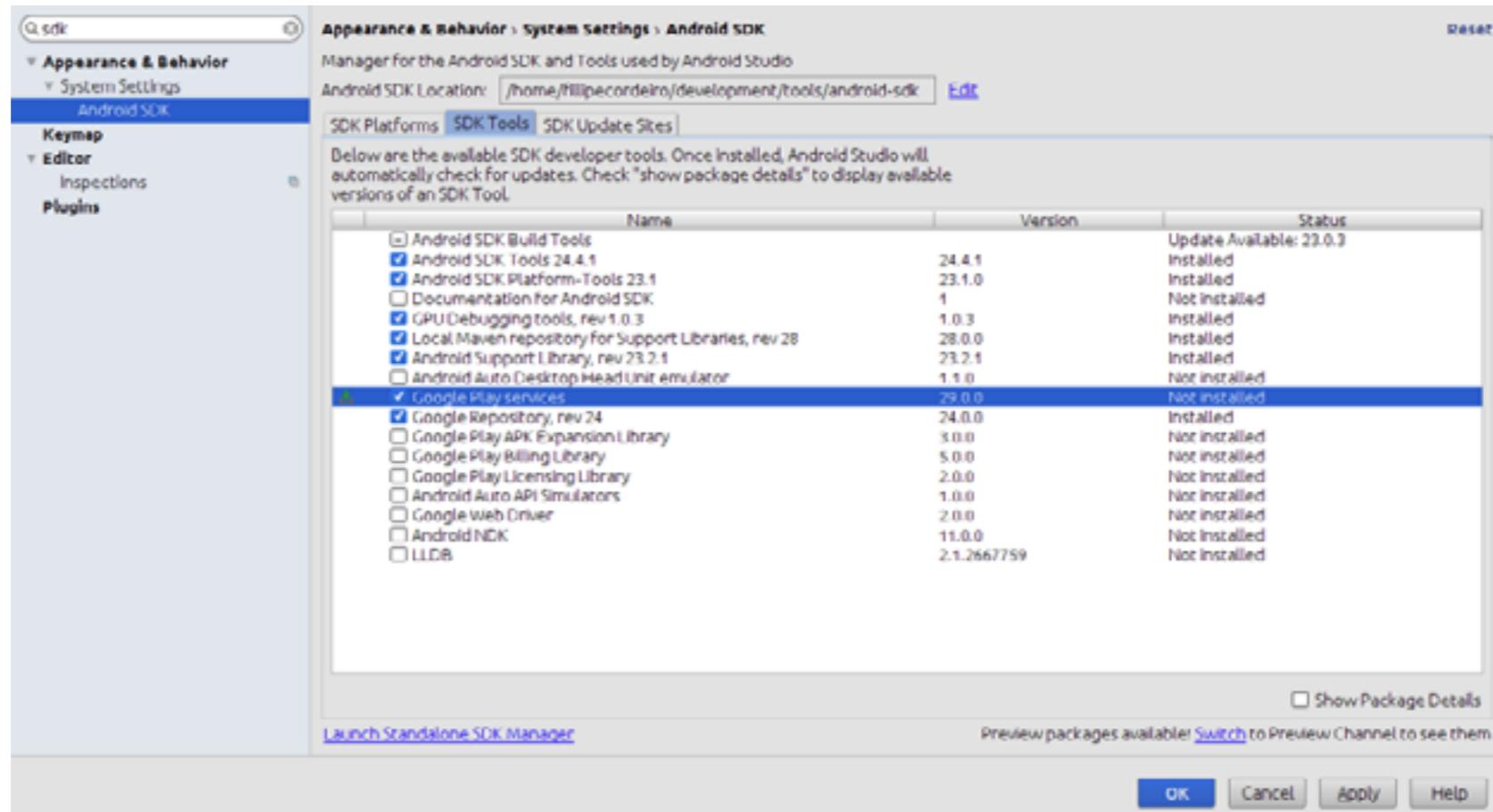
Siga os passos abaixo para fazer a instalação e configuração das bibliotecas.

1. Abra o **Android Studio** e inicie o Android SDK Manager em **File > Settings > Appearance & Behavior > Android SDK**. Na aba **SDK Platforms** habilite a opção **Show Package Details** para serem mostrados todas as opções de cada plataforma. Selecione as opções do Google APIs dentro de qualquer um dos níveis de API 17 ou maior.





2. Agora, na aba SDK Tools selecione a opção Google Play Services e clique no botão Apply.

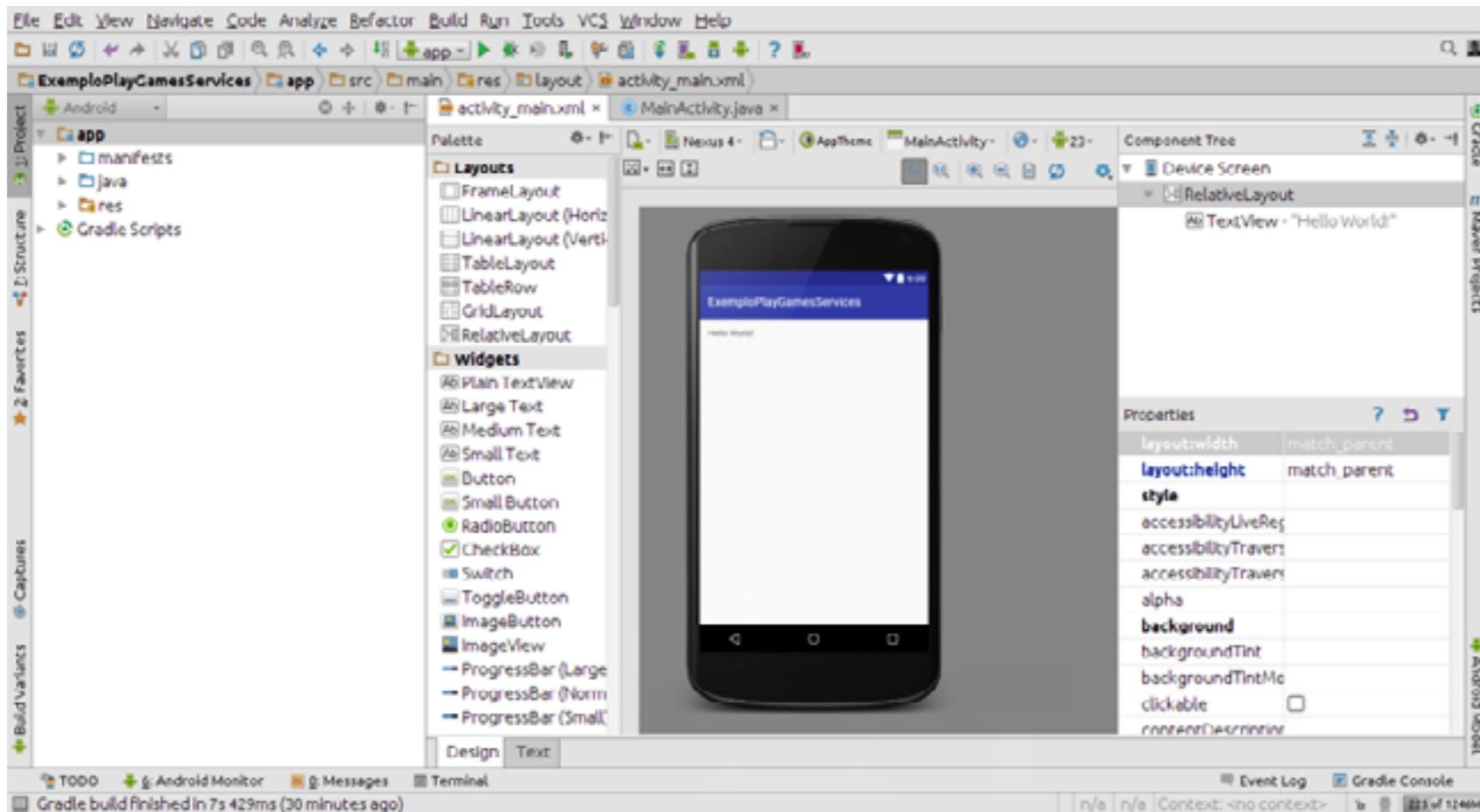


Agora você pode começar a utilizar os recursos do Play Games Services.



Iniciando o projeto

Crie um novo projeto no Android Studio escolhendo a versão 15 minima de API e o tipo de **Activity** sendo **Empty Activity**. Dessa forma iniciamos com um projeto totalmente vazio e mais fácil de manipular.





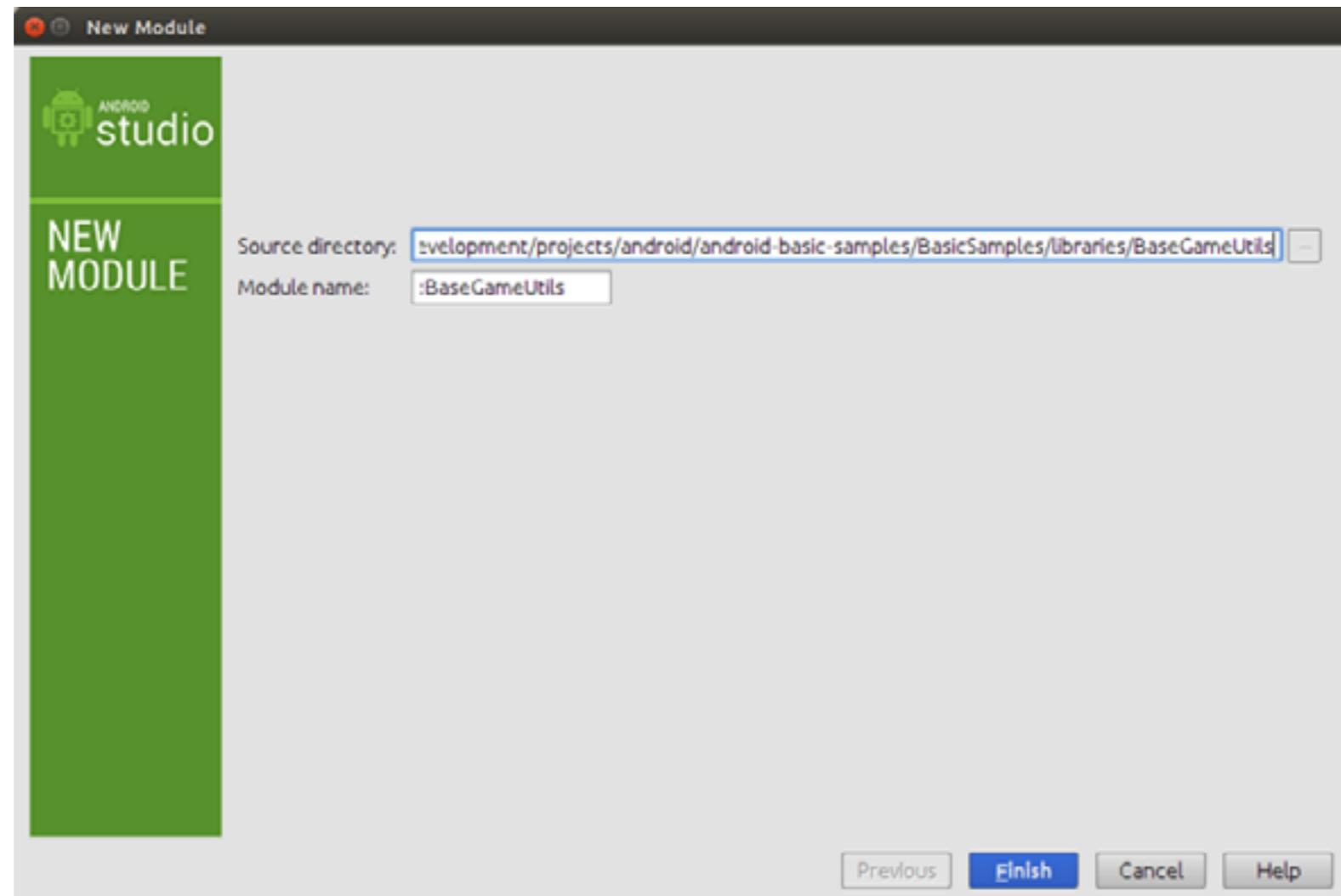
Para facilitar o desenvolvimento do nosso projeto, nós vamos utilizar um projeto base fornecido pelo Google com algumas funcionalidades pré-configuradas.

Faça o download do projeto exemplo [nesta página](#). Você pode utilizar o Git para clonar o código fonte ou fazer o download do pacote ZIP clicando no botão Download ZIP (extraia os arquivos).

Agora precisamos importar a biblioteca **BaseGameUtils** que veio junto com o projeto exemplo do Google.



Com o projeto aberto no Android Studio, vá em **File > New > Import Module**. Selecione a biblioteca que está dentro de **android-basic-samples/BasicSamples/libraries/BaseGameUtils** e clique em **Finish**.





Adicione o projeto **BaseGameUtils** como dependência no arquivo **build.gradle** do módulo **app**.

```
dependencies {  
    compile project(':BaseGameUtils')  
}
```

No arquivo **build.gradle** do módulo **BaseGameUtils** adicione a seguinte configuração dentro de **android {...}**.

```
defaultConfig {  
    minSdkVersion 15  
}
```

Agora precisamos configurar nosso aplicativo e o Google Play Services dentro do projeto.



Crie o arquivo **res/values/ids.xml**, é aqui que vamos guardar algumas informações do nosso jogo. Adicione o conteúdo abaixo.

```
<resources>  
    <string name="app_id">ID_DO_SEU_APP</string>  
</resources>
```

Adicione o ID do aplicativo e a versão do Google Play Services no **AndroidManifest.xml**.

```
<meta-data  
    android:name="com.google.android.gms.games.APP_ID"  
    android:value="@string/app_id" />  
  
<meta-data  
    android:name="com.google.android.gms.version"  
    android:value="@integer/google_play_services_version" />
```

Pronto, podemos começar a codificar nosso jogo.



Fazendo login na conta do Google

Quando você utiliza os serviços do Google em seus aplicativos Android, você precisa login com seu usuário em sua conta do Google.

Existem várias formas de implementar isso, mas nós vamos aproveitar uma funcionalidade pronta usando a classe **BaseGameActivity** juntamente com os botões padrão de login e logout.

Abra o arquivo de layout **activity_main.xml** da sua aplicação e adicione os botões de login e logout.

```
<!-- sign-in button -->  
  
<com.google.android.gms.common.SignInButton  
    android:id="@+id/sign_in_button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content" />  
  
<!-- sign-out button -->  
  
<Button  
    android:id="@+id/sign_out_button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Logout"  
    android:visibility="gone" />
```



Altere a sua **MainActivity** e estenda ela á **BaseGameActivity**. Dessa forma podemos automatizar certas partes do processo de login dos usuários. Vamos também implementar a interface **View.OnClickListener** para usarmos mais para frente.

```
public class MainActivity extends BaseGameActivity implements  
View.OnClickListener
```

Agora vamos recuperar as referências dos botões no onCreate e adicionar nossa MainActivity como um listener.

```
findViewById(R.id.sign_in_button).setOnClickListener(this);  
findViewById(R.id.sign_out_button).setOnClickListener(this);
```



Vamos configurar os eventos onClick nos botões para responderem a alguma ação:

```
@Override  
public void onClick(View view) {  
    if (view.getId() == R.id.sign_in_button){  
        beginUserInitiatedSignIn();  
    }  
    else if (view.getId() == R.id.sign_out_button){  
        signOut();  
        findViewById(R.id.sign_in_button).setVisibility(View.VISIBLE);  
        findViewById(R.id.sign_out_button).setVisibility(View.GONE);  
    }  
}
```

Nós vamos usar os métodos da classe **BaseGameActivity** para efetuar o login (beginUserInitiatedSignIn e signOut). Quando o aplicativo é iniciado, ele tentará fazer o login automaticamente e também usará os botões de login e logout.



Agora precisamos implementar os dois métodos de callback em nossa Activity:

```
@Override  
public void onSignInSucceeded(){  
    findViewById(R.id.sign_in_button).setVisibility(View.GONE);  
    findViewById(R.id.sign_out_button).setVisibility(View.VISIBLE);  
}
```

```
@Override  
public void onSignInFailed(){  
    findViewById(R.id.sign_in_button).setVisibility(View.VISIBLE);  
    findViewById(R.id.sign_out_button).setVisibility(View.GONE);  
}
```

Você pode adicionar mais código se for necessário. Você também pode optar por salvar o progresso do jogador, mesmo se ele não estiver logado, mas isso depende do seu jogo.

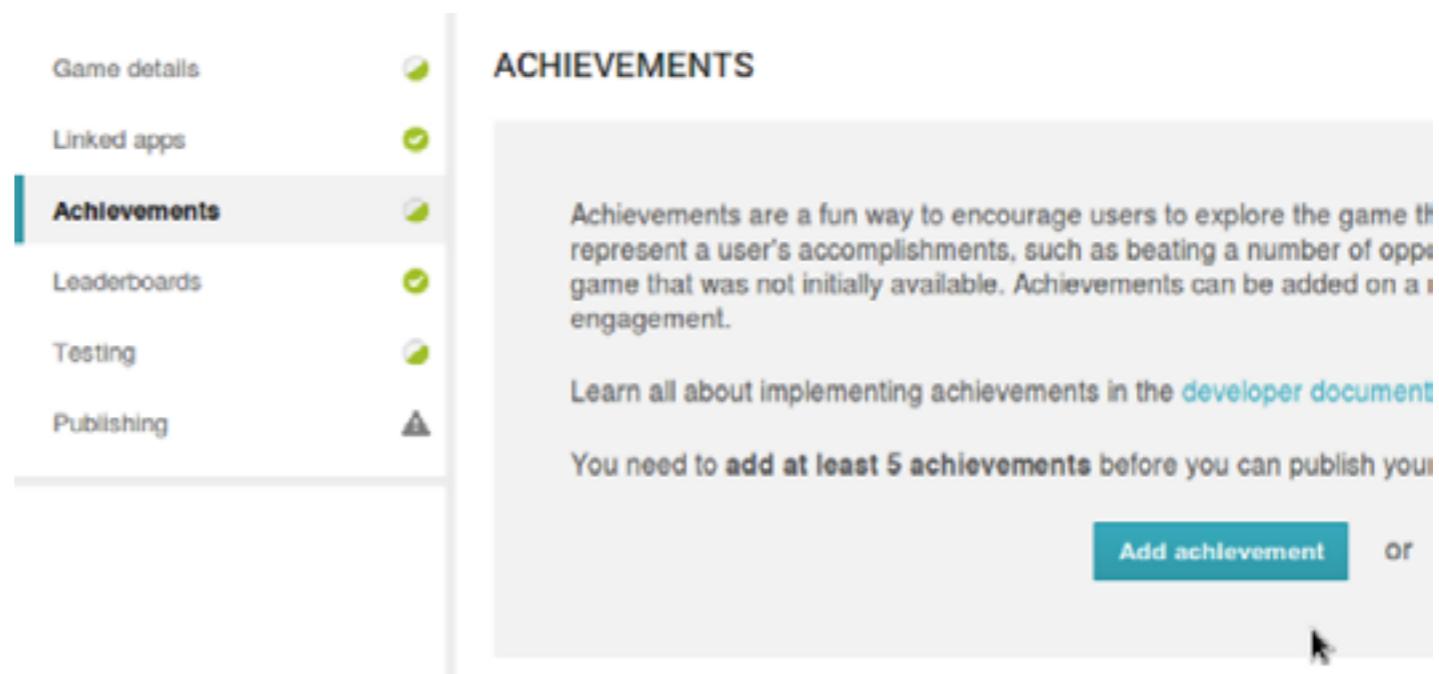


Trabalhando com Achievements

Agora precisamos implementar o sistema de achievements para que você possa desbloquear as conquistas quando o jogador atingir os objetivos. Naturalmente, isso vai depender da finalidade do seu próprio jogo, mas neste caso vamos implementar de forma simples em nosso aplicativo de exemplo.

Criando um Achievement

Acesse seu jogo no Google Developer Console, clique no botão **Achievements** do lado esquerdo do menu e clique em **Add achievement**.





[New Achievement](#) [Save](#) [Save and add another achievement](#)

English (United States) – en-US

Name
English (United States) – en-US

Description
English (United States) – en-US
(optional for testing)

Icon [?](#)
512 x 512
png or jpg
(optional for testing)

Numero Correto
14 of 100 characters

Acertou o numero
16 of 500 characters



Na página de criação de um achievement preencha nome, descrição e um ícone para o seu achievement e escolha um estado, pontos e a ordem da lista. Clique em **Save**.

Copie o ID do achievement que você vê na listagem.

Implementando um Achievement

Vamos começar com o layout, que inclui os botões de login/logout adicionados anteriormente:

```
<LinearLayout
    android:id="@+id/sign_in_buttons"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:orientation="horizontal">

    <!-- sign-in button -->
    <com.google.android.gms.common.SignInButton
        android:id="@+id/sign_in_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <!-- sign-out button -->
    <Button
        android:id="@+id/sign_out_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

<!-- show achievements -->
<Button
    android:id="@+id/show_achievements"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Achievements" />

</LinearLayout>

<RelativeLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/sign_in_buttons">

    <TextView
        android:id="@+id/guess_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:text="Acerto o número!"
        android:textSize="30sp"
        android:textStyle="bold" />

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/guess_text"
    android:orientation="vertical">

    <Button
        android:id="@+id/btn7"
        android:layout_width="0dp"
```



```
    android:layout_height="match_parent"
    android:layout_margin="1dp"
    android:layout_weight="1"
    android:background="#ff000033"
    android:gravity="center"
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="7"
    android:text="7"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />
```

```
<Button
    android:id="@+id	btn8"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_margin="1dp"
    android:layout_weight="1"
    android:background="#ff000033"
    android:gravity="center"
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="8"
    android:text="8"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />

<Button
```

```
    android:id="@+id	btn9"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_margin="1dp"
    android:layout_weight="1"
    android:background="#ff000033"
    android:gravity="center"
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="9"
    android:text="9"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />
```

```
</LinearLayout>
```

```
<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:orientation="horizontal">
```

```
    <Button
        android:id="@+id	btn4"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:background="#ff000033"
        android:gravity="center"
```

```
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="4"
    android:text="4"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />
```

```
<Button
    android:id="@+id	btn5"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_margin="1dp"
    android:layout_weight="1"
    android:background="#ff000033"
    android:gravity="center"
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="5"
    android:text="5"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />
```

```
<Button
    android:id="@+id	btn6"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_margin="1dp"
    android:layout_weight="1"
```



```
    android:background="#ff000033"
    android:gravity="center"
    android:onClick="btnPressed"
    android:padding="5dp"
    android:tag="6"
    android:text="6"
    android:textColor="#ffffffff"
    android:textSize="30sp"
    android:textStyle="bold" />

</LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:orientation="horizontal">

    <Button
        android:id="@+id(btn1"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:background="#ff000033"
        android:gravity="center"
        android:onClick="btnPressed"
        android:padding="5dp"
        android:tag="1"
        android:text="1"
        android:textColor="#ffffffff"
        android:textSize="30sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id(btn2"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:background="#ff000033"
        android:gravity="center"
        android:onClick="btnPressed"
        android:padding="5dp"
        android:tag="2"
        android:text="2"
        android:textColor="#ffffffff"
        android:textSize="30sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id(btn3"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_margin="1dp"
        android:layout_weight="1"
        android:background="#ff000033"
        android:gravity="center"
        android:onClick="btnPressed"
        android:padding="5dp"
        android:tag="3"
        android:text="3"
        android:textColor="#ffffffff"
        android:textSize="30sp"
        android:textStyle="bold" />

</LinearLayout>
```



```
<Button  
    android:id="@+id	btnAgain"  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_margin="1dp"  
    android:layout_weight="1"  
    android:background="#ffff00"  
    android:enabled="false"  
    android:gravity="center"  
    android:onClick="btnPressed"  
    android:padding="5dp"  
    android:tag="-1"  
    android:text="Again"  
    android:textColor="#ffff00"  
    android:textSize="30sp"  
    android:textStyle="bold" />  
  
    </LinearLayout>  
    </LinearLayout>  
    </RelativeLayout>
```

Nesse layout nós colocamos um botão de achievements e os botões de **login** e **logout**. Não vou entrar em muitos detalhes sobre o jogo de exemplo.

O jogo seleciona um número aleatório entre 0 e 9 e o jogador tem que escolher um número para tentar adivinhar o número selecionado.

O jogo atualiza o campo de texto avisando se o usuário adivinhou ou não o número. Se o palpite for correto, um achievement é desbloqueado.



Volte para sua **MainActivity** e adicione as seguintes variáveis em escopo de classe:

```
private Button button0, button1, button2, button3, button4, button5,  
        button6, button7, button8, button9, buttonAgain;  
private int number;  
private Random rand;  
private TextView info;
```



Estes botões representam os números, o gerador de números aleatórios e o campo de texto. Adicione o seguinte código ao seu método **onCreate**:

```
findViewById(R.id.show_achievements).setOnClickListener(this);
```

```
button0=(Button)findViewById(R.id.btn0);
button1=(Button)findViewById(R.id.btn1);
button2=(Button)findViewById(R.id.btn2);
button3=(Button)findViewById(R.id.btn3);
button4=(Button)findViewById(R.id.btn4);
button5=(Button)findViewById(R.id.btn5);
button6=(Button)findViewById(R.id.btn6);
button7=(Button)findViewById(R.id.btn7);
button8=(Button)findViewById(R.id.btn8);
button9=(Button)findViewById(R.id.btn9);
buttonAgain=(Button)findViewById(R.id.btnAgain);
```

```
info=(TextView)findViewById(R.id.guess_text);
rand=new Random();
number=rand.nextInt(10);
```



Agora adicione um método chamado **disableNumbers**, que chamaremos quando o usuário fizer a escolha de um número:

```
private void disableNumbers() {
    button0.setEnabled(false);
    button0.setTextColor(Color.parseColor("#ff000033"));
    button1.setEnabled(false);
    button1.setTextColor(Color.parseColor("#ff000033"));
    button2.setEnabled(false);
    button2.setTextColor(Color.parseColor("#ff000033"));
    button3.setEnabled(false);
    button3.setTextColor(Color.parseColor("#ff000033"));
    button4.setEnabled(false);
    button4.setTextColor(Color.parseColor("#ff000033"));
    button5.setEnabled(false);
    button5.setTextColor(Color.parseColor("#ff000033"));
    button6.setEnabled(false);
    button6.setTextColor(Color.parseColor("#ff000033"));
    button7.setEnabled(false);
    button7.setTextColor(Color.parseColor("#ff000033"));
    button8.setEnabled(false);
    button8.setTextColor(Color.parseColor("#ff000033"));
    button9.setEnabled(false);
    button9.setTextColor(Color.parseColor("#ff000033"));
    buttonAgain.setEnabled(true);
    buttonAgain.setTextColor(Color.parseColor("#ff000033"));
}
```



Implemente outro método chamado **enableNumbers**, que é invocado quando o usuário escolheu jogar novamente:

```
private void enableNumbers(){
    button0.setEnabled(true);
    button0.setTextColor(Color.WHITE);
    button1.setEnabled(true);
    button1.setTextColor(Color.WHITE);
    button2.setEnabled(true);
    button2.setTextColor(Color.WHITE);
    button3.setEnabled(true);
    button3.setTextColor(Color.WHITE);
    button4.setEnabled(true);
    button4.setTextColor(Color.WHITE);
    button5.setEnabled(true);
    button5.setTextColor(Color.WHITE);
    button6.setEnabled(true);
    button6.setTextColor(Color.WHITE);
    button7.setEnabled(true);
    button7.setTextColor(Color.WHITE);
    button8.setEnabled(true);
    button8.setTextColor(Color.WHITE);
    button9.setEnabled(true);
    button9.setTextColor(Color.WHITE);
    buttonAgain.setEnabled(false);
    buttonAgain.setTextColor(Color.parseColor("#ffff00"));
}
```



No arquivo **ids.xml** que criamos anteriormente, adicione a seguinte entrada com o ID de sua achievement.

```
<string name="resposta_certa_achievement">Cgklip-mysUTEAIQAw</string>
```

Agora adicione o método que estabelecemos como atributo **onClick** para os botões:

```
public void btnPressed(View v) {
    int btn = Integer.parseInt(v.getTag().toString());
    if (btn < 0) {
        number = rand.nextInt(10);
        enableNumbers();
        info.setText("Acerte o número!");
    } else {
        if (btn == number) {
            info.setText("Acertou! O número era " + number);
            if (getApiClient().isConnected())
                Games.Achievements.unlock(getApiClient(),
                    getString(R.string.resposta_certa_achievement));
        } else {
            info.setText("Errou! O número era " + number);
        }
        disableNumbers();
    }
}
```



Nós chamamos o Games.Achievements se o palpite do usuário for correto passando o ID do achievement que configuramos em nosso **ids.xml**.

Por último, vamos permitir que o usuário visualize seus achievements do jogo. Isso acontecerá quando ele clicar no botão **Achievement**. No código do método **onClick** adicione um **else if**:

```
else if (view.getId() == R.id.show_achievements){  
    startActivityForResult(Games.Achievements.getAchievementsIntent(  
        getApiClient()), 1);  
}
```

Usamos o método **getAchievementsIntent** com um número inteiro para exibir os achievements dos usuários dentro do jogo.



Trabalhando com Leaderboards

Depois de definirmos e implementarmos os achievements do nosso jogo, precisamos desenvolver nosso leaderboard para que nosso aplicativo Android possa mostrar as tabelas de classificação, gravar a pontuação do jogador e comparar o placar contra a pontuação do jogador em sessões anteriores do jogo.

Criando um Leaderboard

Acesse seu jogo no Google Developer Console, clique no botão Leaderboards do lado esquerdo do menu e clique em **Add leaderboard**.

LEADERBOARDS

Leaderboards allow users to compare their scores in your game. players to keep playing and do better.

Learn all about implementing leaderboards in the [developer docs](#).

[Add leaderboard](#)





Tenha certeza que você entendeu o conceito de **Leaderboard** no Play Games Services, isso é muito importante para continuarmos.

Você pode fazer um monte de coisas diferentes utilizando os Leaderboards, por isso esse exemplo é apenas um ponto de partida.

Digite os detalhes do seu novo leaderboard. Para o exemplo, usamos o nome **Menos Chutes** e selecionamos **Smaller is Better** na seção **Ordering**.



Adicione um ícone se quiser, se não uma imagem padrão será usada. Salve sua nova leaderboard e copie seu ID.

[←](#) NEW LEADERBOARD [Save](#) [Save and add another leaderboard](#)

English (United States) – en-US

Name
English (United States) – en-US

Menos Chutes
12 of 100 characters

Score formatting

Numeric ▾

Number of decimal places:
0 ▾

What it will look like:
123,450,000

Add custom unit [?](#)

Icon [?](#)
512 × 512
png or jpg
(optional)



If no icon is provided, a standard leaderboard icon will be shown to the users.

Ordering

Larger is better [Smaller is better](#) 



Implementando um Leaderboard

Para essa seção, nós vamos aproveitar todo o código que fizemos anteriormente quando implementamos os achievements.

Vamos começar guardando a informação do nosso leaderboard criado. Abra o arquivo ids.xml que criamos anteriormente e adicione a seguinte entrada com o ID do seu leaderboard.

```
<string name="menos_chutes_leaderboard">Cgklip-mysUTEAIQBQ</string>
```



Agora precisamos adicionar um botão para o jogador acessar o seu leaderboard e acompanhar sua pontuação. Para isso, abra o arquivo **activity_main.xml** e adicione o botão **Leaderboard** abaixo do botão **Achievements**.

```
<Button  
    android:id="@+id/show_leaderboard"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Leaderboard" />
```



Agora precisamos trabalhar com esse botão em nossa **MainActivity**. Você vai precisar que o seguinte código seja adicionado no método **onCreate**.

```
findViewById(R.id.show_leaderboard).setOnClickListener(this);
```

Adicione também um atributo de classe chamado **numGuesses**.

```
private int numGuesses=0;
```

Esse atributo vai controlar o número de vezes que o jogador tentou adivinhar o número correto.



Precisamos mexer também no método **onClick** utilizado pelas teclas numéricas no layout.

Quando o jogador acertar um número além de desbloquearmos seu achievement também vamos submeter seus pontos para o leaderboard.

Veja como ficou a implementação do método **onClick** utilizando o atributo **numGuesses** e fazendo a chamada para o leaderboard.

```
public void btnPressed(View v) {
    int btn = Integer.parseInt(v.getTag().toString());
    if (btn < 0) {
        numGuesses=0;
        number = rand.nextInt(10);
        enableNumbers();
        info.setText("Aerte o número!");
    } else {
        numGuesses++;
        if (btn == number) {
            info.setText("Acertou! O número era " + number);
            if (getApiClient().isConnected()) {
                Games.Achievements.unlock(getApiClient(),
                    getString(R.string.resposta_certa_achievement));
                Games.Leaderboards.submitScore(getApiClient(),
                    getString(R.string.menos_chutes_leaderboard),
                    numGuesses);
            }
        } else if(numGuesses==5) {
            info.setText("Errou! O número era " + number);
            disableNumbers();
        } else {
            info.setText("Tente de novo!");
        }
    }
}
```



Se o jogador clicar no botão **De Novo**, nós zeramos o atributo **numGuesses**. Se o usuário clica em um botão numérico, incrementamos o **numGuesses**.

Então enviamos a pontuação para o leaderboard quando o usuário adivinha o número. O usuário pode fazer até cinco chutes.

O importante aqui é o **submitScore**. Passamos o número de chutes que o jogador levou para acertar o número correto. Se o número de tentativas é menor do que qualquer entrada existente na leaderboard do usuário, sua pontuação será substituída pelo novo valor.

Antes de finalizar nossa implementação, vamos permitir que o usuário exiba o leaderboard do jogo clicando no botão **Leaderboard** que adicionamos.



Veja como ficou o código do método **onClick**.

```
@Override  
public void onClick(View view){  
    if (view.getId() == R.id.sign_in_button){  
        beginUserInitiatedSignIn();  
    } else if (view.getId() == R.id.sign_out_button){  
        signOut();  
        findViewById(R.id.sign_in_button).setVisibility(View.VISIBLE);  
        findViewById(R.id.sign_out_button).setVisibility(View.GONE);  
    } else if (view.getId() == R.id.show_achievements){  
        startActivityForResult(Games.Achievements.getAchievementsIntent(  
            getClient(), 1);  
    } else if (view.getId() == R.id.show_leaderboard){  
        startActivityForResult(Games.Leaderboards.getLeaderboardIntent(  
            getClient(), getString(R.string.menos_chutes_leaderboard)),  
            2);  
    }  
}
```

Isso permitirá que o usuário veja a classificação atual dentro do **leaderboard**.



Plataformas e Game Engines

O foco deste ebook é mostrar os serviços do **Play Games Services** na plataforma Android como exemplo, mas existem várias outras plataformas e engines de jogos que é possível utilizar com o serviço.

O objetivo do Google é facilitar e tornar seu jogo viável em várias plataformas diferentes centralizando os dados em um lugar só, na nuvem. Por essa razão existem diversas SDKs e plugins para diferentes tecnologias.

Você pode encontrar todas as informações entrando [no site oficial](#).



Plataformas

Hoje é possível desenvolver diretamente utilizando as APIs do serviço para Android, iOS, Web e C++. Apenas para a Web não existe uma SDK de desenvolvimento, mas todos os serviços são em formato **REST**, o que facilita muito. Veja a tabela com as **SDKs** disponíveis:

Plataforma	SDK	Descrição
Android	Google Play Services SDK	Biblioteca em Java.
iOS	Play Games C++ SDK e Google Sign-In iOS SDK	Bibliotecas em C++ e Objective-C.
C++	Play Games C++ SDK	Biblioteca em C++
Web	REST API	Serviços em formato REST.



Game Engines

Todos sabemos que os desenvolvedores de jogos utilizam engines que auxiliam no desenvolvimento de games. Por essa razão, o Google disponibiliza alguns plugins para engines conhecidas no mercado que facilitam a utilização do serviço. Veja na tabela abaixo os plugins disponíveis para cada engine.

Engine	Plataforma	Linguagem
Unity	Android, iOS	C#
Apportable	Android, iOS	Objective-C
Corona	Android	Lua
GameMaker Studio	Android	GML
LibGDX	Android	Java
Marmalade	Android	C/C++
PlayCanvas	Web	JavaScript



Conclusão

Espero que este ebook tenha aberto as portas para você para o Play Games Services.

Hoje a tecnologia em nuvem é uma parte muito importante da maioria dos aplicativos e jogos de sucesso que temos no mercado.

Vimos como funciona a Google Play Games, a plataforma do Google onde os jogadores podem ver os dados de seus jogos salvos, estatísticas de conquistas e também encontrar outros jogadores para jogarem juntos.

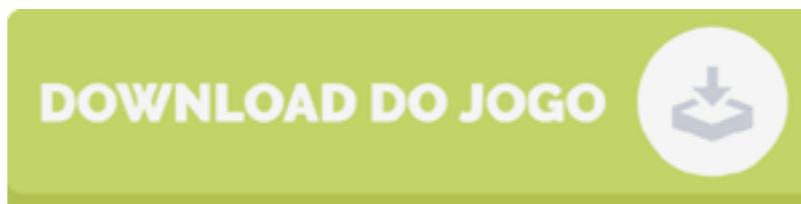
Entendemos também como é poderoso o serviço do Play Games Services facilitando muito o desenvolvimento de nossos jogos trazendo conceitos em forma de serviços e guardando os dados em nuvem de forma segura e distribuída.

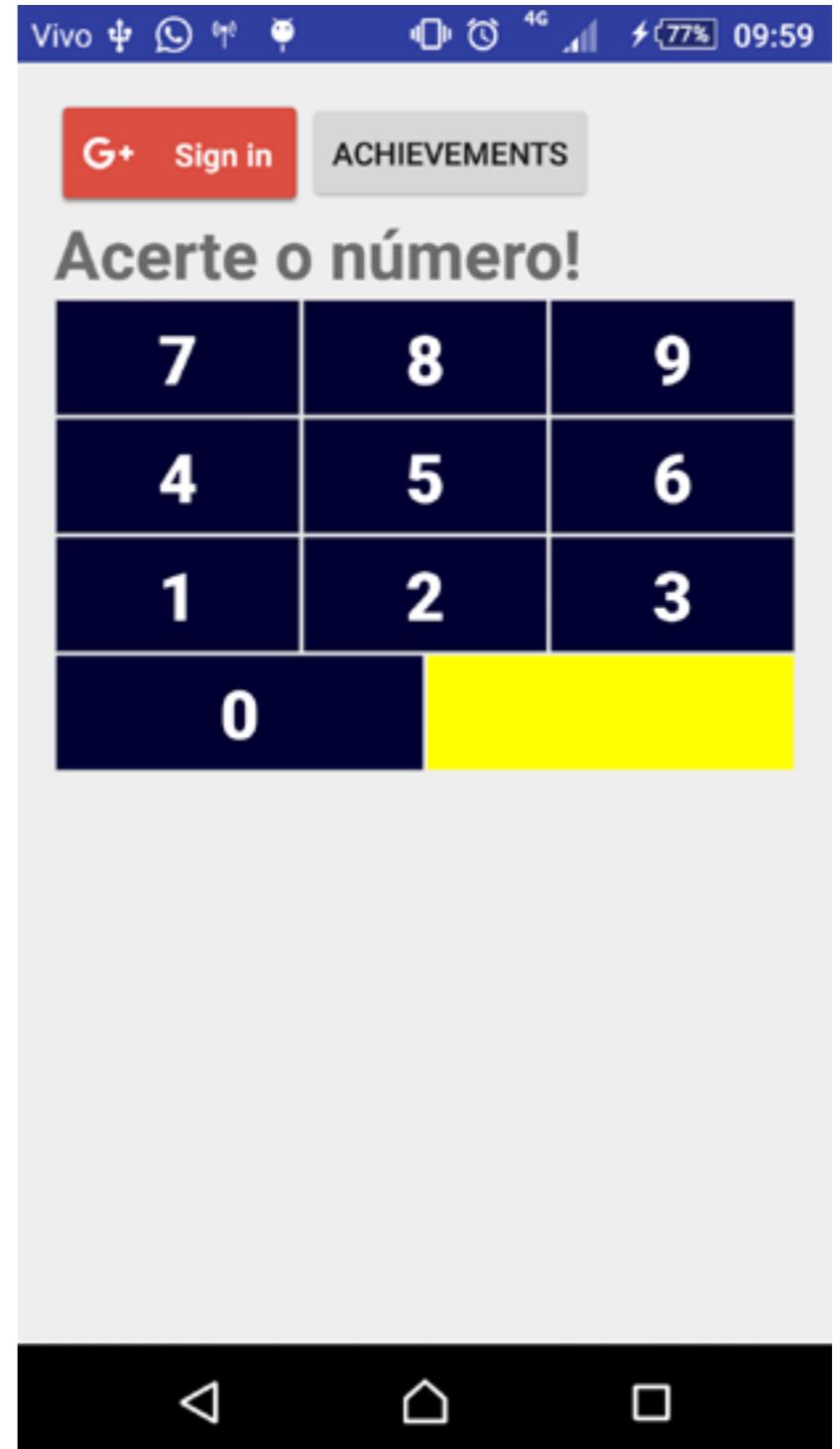
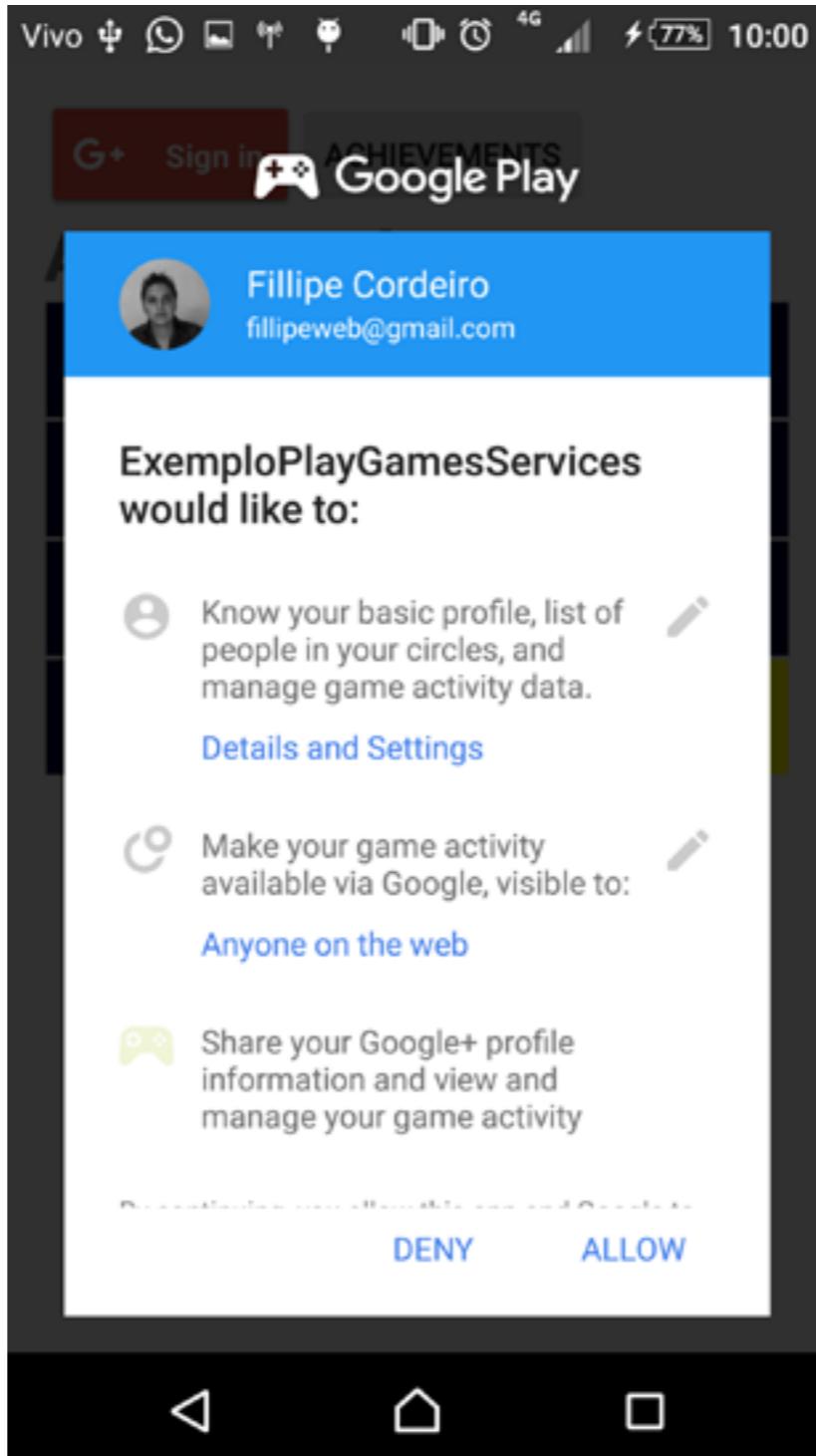
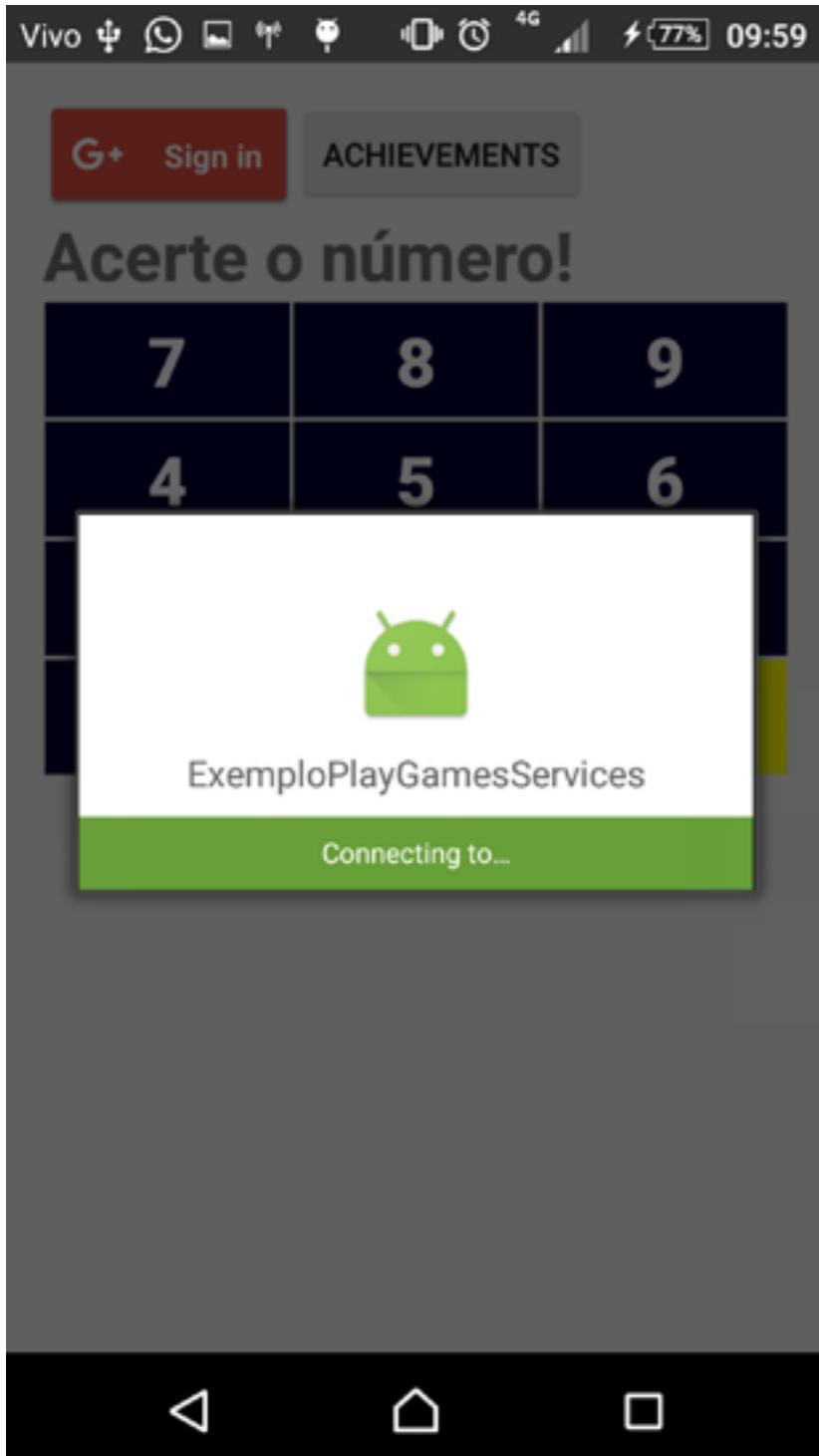


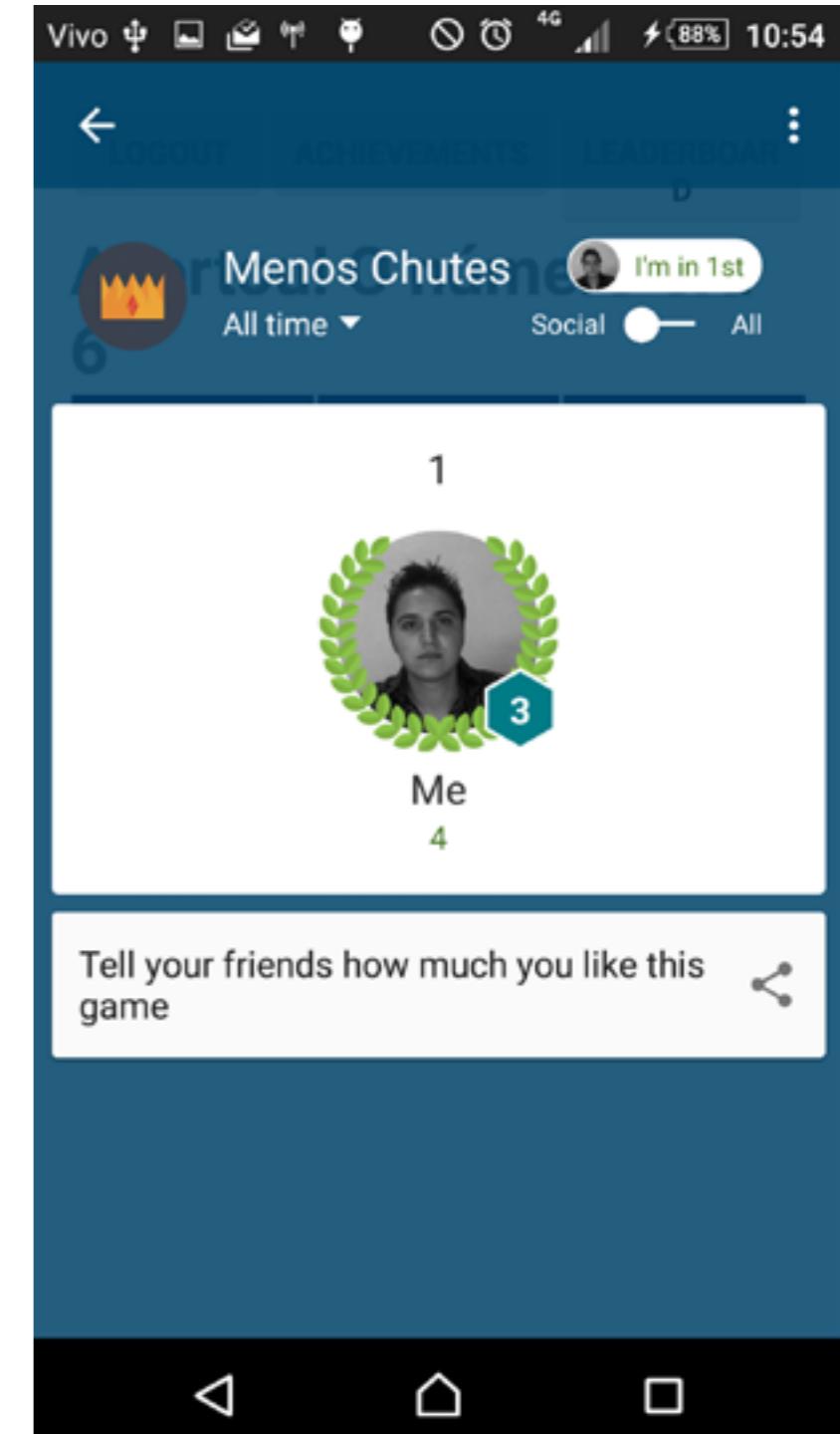
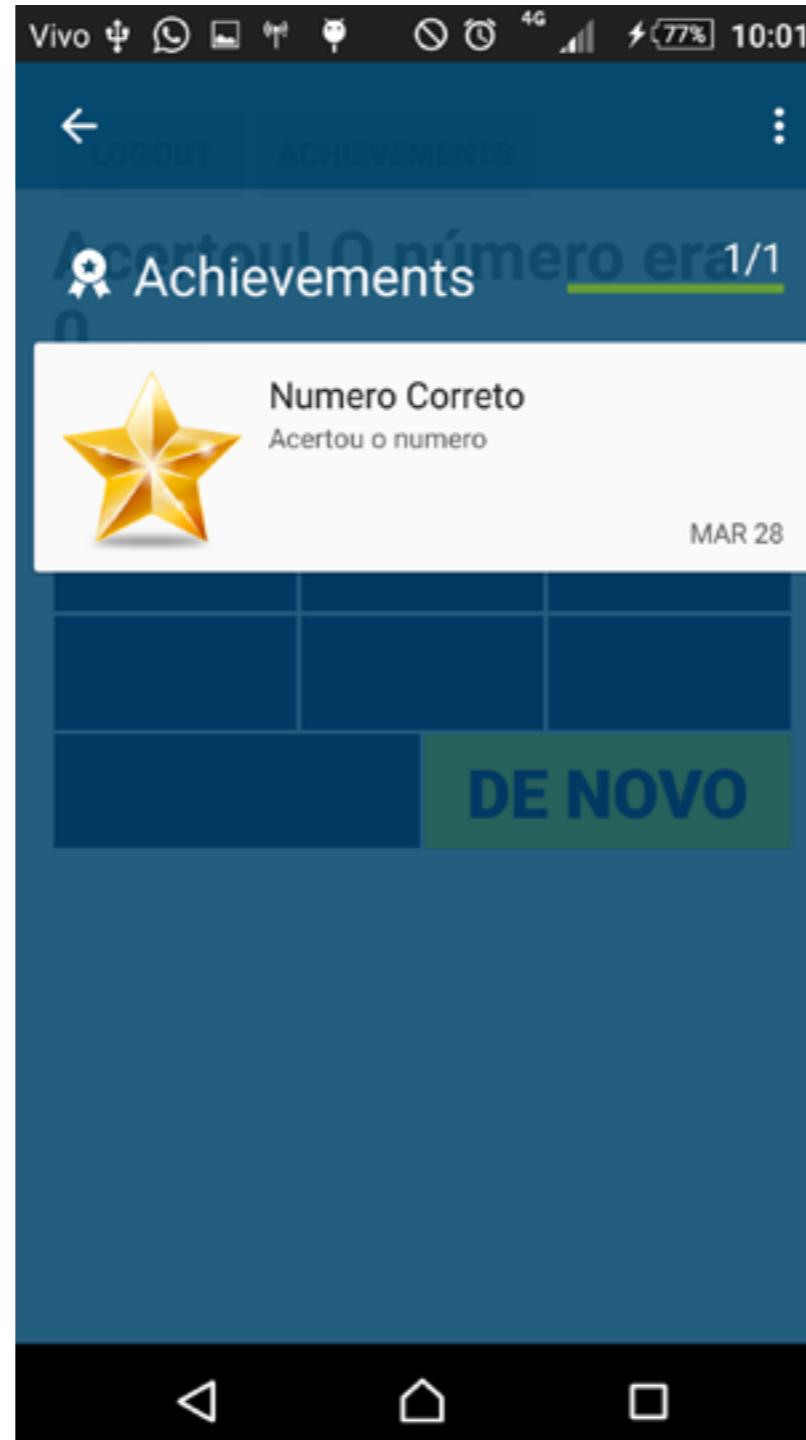
Vários conceitos importantes no desenvolvimento de jogos estão disponíveis no serviço permitindo que o desenvolvedor não se preocupe tanto com eles, mas sim, em criar um jogo que entretenha o usuário e divirta as pessoas.

Esses conceitos podem ser facilmente implementados com poucas linhas de código e algumas configurações.

De forma prática, desenvolvemos um jogo de exemplo totalmente funcional usando dois serviços do Play Games Services, achievements e leaderboards.









O objetivo do jogo era adivinhar um número secreto aleatório em no máximo 5 tentativas, caso o jogador acerte, era liberado um achievement para e suas estatísticas eram enviadas para seu leaderboard.

Utilizando os dois botões que criamos, foi possível visualizar todas os achievements do jogador e também o seu leaderboard para acompanhar o progresso geral dentro do jogo.

Finalizando...

Espero que tenha gostado desse material e que tenha sido muito útil para você, nosso objetivo era apresentar a tecnologia e te dar conhecimento o suficiente para utilizá-la e dar muito mais qualidade aos seus jogos.

Grande abraço da equipe AndroidPro e Produção de Jogos.



O objetivo do Produção de Jogos é ajudar pessoas a trilharem o caminho completo do desenvolvimento de jogos, da concepção da ideia à publicação e venda do jogo.

Produzimos conteúdo sobre carreira, desenvolvimento, mercado e marketing de jogos digitais, que disponibilizamos através de entrevistas em vídeos, blog posts, ebooks e palestras online.

Tudo isso em um site livre de propagandas e voltado para o leitor. Para saber mais sobre o mundo dos jogos digitais, não deixe de visitar o site Produção de Jogos em: producaodejogos.com.

- *Raphael Dias*

Engenheiro da computação e desenvolvedor de software há mais de 9 anos, com experiência em tecnologias como Java, Python e Android. Já ajudou mais de 15 mil pessoas a mergulhar no universo do Desenvolvimento Android com o blog AndroidPro.

O AndroidPro foi montado com o objetivo de passar conhecimento de Desenvolvimento Android para todos. Além disso, tenho o grande desejo e objetivo de, a partir do AndroidPro, colaborar com o crescimento do Brasil e dos nossos profissionais na área de aplicativos Android.

Por isso, também falamos sobre Carreira, e Mercado de Trabalho com Android. Afinal, o momento que estamos vivendo é ideal para se inserir no mercado mobile, ainda mais no nosso país!

- *Fillipe Cordeiro*

