

# MCZA033 - Programação Avançada para Dispositivos Móveis

## Plano de ensino

Prof. Diogo S. Martins  
Centro de Matemática, Computação e Cognição  
Universidade Federal do ABC

Q2 2019  
v. 08/06

### 1 Informações básicas

- TPI: 0-4-4
- Laboratório: 407-2
- Aulas (Turma A):
  - seg 21-23h 407-2 semanal
  - qui 19-21h 407-2 semanal
- Worksite TIDIA-AE: PADM-2019Q2  
Acesso ao material de apoio, comunicação intra-disciplina e entrega de atividades avaliativas
- Atendimento:
  - Plantão de dúvidas:
    - seg 19-21h 528-2 semanal
  - Comunicação com o professor:
    - \* Prioritariamente via ferramenta Mensagens do Tidia
    - \* Por email (caso Tidia indisponível): [santana.martins@ufabc.edu.br](mailto:santana.martins@ufabc.edu.br)
    - \* Presencialmente, na sala 528-2, no horário de plantão.
- Repositório Bitbucket: [https://bitbucket.org/diogo\\_martins/padm-2019q2/src](https://bitbucket.org/diogo_martins/padm-2019q2/src)  
Acesso ao código-fonte de apoio às aulas

### 2 Descrição da disciplina

O desenvolvimento de aplicações para dispositivos móveis (e.g. smartphones, tablets, dispositivos vestíveis, etc.) demanda o tratamento de requisitos que são inerentes à computação móvel, entre os quais pode-se citar: *i*) tamanho reduzido de tela; *ii*) meios de interação limitados para entrada e saída de dados; *iii*) funcionamento restrito à duração de bateria; *iv*) conectividade potencialmente intermitente com o deslocamento geográfico; *v*) poder de processamento inferior a outros dispositivos de computação pessoal contemporâneos (e.g. desktops, notebooks, etc.); *vi*) utilização em múltiplos contextos no cotidiano de seus usuários; entre outros. Com vistas a essas particularidades, a disciplina trata dos principais aspectos do desenvolvimento para dispositivos móveis. Em abrangência, aborda o desenvolvimento de aplicações com atenção aos requisitos específicos de computação móvel, ao mesmo tempo em que reforça conceitos de projeto orientado a objetos aplicáveis a outras classes de computação. Em profundidade, promove a verificação e experimentação desses conceitos com base na plataforma Android.

### 3 Requisitos recomendados

Para participar dessa disciplina é recomendado ter cursado e sido aprovado em:

- BC1424 - Algoritmos e Estruturas de Dados I
- BC1501 - Programação Orientada a Objetos
- MC0037 - Programação para Web

### 4 Objetivos

- Familiarizar-se com os fundamentos do desenvolvimento de aplicações para dispositivos móveis;
- Ser capaz de projetar aplicações para dispositivos móveis utilizando e refinando padrões de projeto;
- Ser capaz de desenvolver aplicações que usem adequadamente os recursos dos dispositivos móveis.

Ao final do curso, espera-se que o aluno, aprovado com conceito satisfatório, possua habilidades que permitam, a partir de um conjunto de requisitos, projetar aplicações para dispositivos móveis que sejam eficientes, confiáveis, seguras e de fácil manutenção.

### 5 Ementa

- Conceitos fundamentais de programação móvel;
- Projeto de interfaces para aplicações móveis;
- Armazenamento de dados para aplicações móveis;
- Integração com recursos dos dispositivos;
- Integração com outros sistemas: comunicação cliente-servidor;
- Desenvolvimento para dispositivos heterogêneos (smartphones vs. tablets).

### 6 Bibliografia

- Literatura da plataforma Android. URL: (<http://developer.android.com/develop/index.html>). Acesso: Jan. 2019.
- Medneiks, Zigurd; Dornin, Laird; Meike, G. Blake; Nakamura, Masumi. “Programando o Android”. Editora Novatec, 2a edição, 2012.
- Bill Phillips; Brian Hardy. *Android Programming: The Big Nerd Ranch Guide*. Big Nerd Ranch Guides. 2a. edição. 2015.
- Mark L. Murphy. *The Busy Coder’s Guide to Android Development*. CommonsWare. Edição 7.6. 2016.
- Therese Neil. *Mobile Design Pattern Gallery*. O’Reilly. 2nd edition. 2014.
- Brian Fling. *Mobile Design and Development*. O’Reilly. 2009.

## 7 Metodologia de ensino-aprendizagem

A metodologia fundamental dessa disciplina é a aprendizagem ativa<sup>1</sup>, segundo a qual a responsabilidade pelo aprendizado é do aprendiz. Essa metodologia enfatiza que as atividades em aula devem estimular a análise, a síntese e a avaliação de conteúdos. Difere em grande medida da aprendizagem passiva, na qual as atividades em aula geralmente demandam apenas o consumo passivo de conteúdos de palestras, vídeos ou leituras. Os métodos de ensino-aprendizagem que empregaremos são a *aula invertida*<sup>2</sup> e a *aprendizagem baseada em problemas*<sup>3</sup>.

O tempo de aula será usado essencialmente para a resolução de problemas relacionados ao conteúdo abordado. Na maior parte dos casos, os problemas envolverão a construção de uma aplicação com base em um conjunto de requisitos. Naturalmente, não podemos ignorar os aspectos teóricos do conteúdo, portanto utilizaremos o método da aula invertida — cada tema constitui um ciclo, e cada ciclo é formado pelas seguintes atividades:

- **Estudo prévio:** alunos estudam os conteúdos do ciclo, com base nos recursos indicados pelo professor;
- **Avaliação formativa:** antes da aula, é necessário responder um questionário de verificação de conhecimento ou propor uma resolução para um problema simples;
- **Prática em sala:** resolução de problemas ou mini-projetos sobre o conteúdo estudado;
- **Exercícios:** resolução de exercícios autonomamente

A orquestração dessas atividades envolve o seguinte roteiro:

1. Os conteúdos (e.g. livro, slides, materiais complementares, etc.) referentes à próxima aula serão divulgados previamente;
2. Aluno deve estudar esses conteúdos antes da próxima aula;
3. Uma atividade de verificação de conhecimento deverá ser efetuada, extra-classe, antes da aula;
4. O aluno vem para a aula, já tendo uma base do conteúdo;
5. Na aula, praticaremos a solução de problemas relacionados ao conteúdo. Algumas aulas poderão incluir atividades avaliativas;
6. Após a aula, o aluno pratica em mais profundidade por meio da resolução de problemas autonomamente.

Essa metodologia difere substancialmente da metodologia tradicional à medida que exige alto comprometimento do aluno. O sucesso das aulas depende da participação dos alunos na solução dos problemas (e.g. via discussão, sugestões, etc.). Não basta ficar na aula somente de “corpo presente”, é preciso participar ativamente. Também é contraproducente deixar para estudar apenas na semana da prova, pois nesse caso falhará em todas as atividades prévias.

### Sobre as avaliações formativas

Uma avaliação formativa consiste em exercícios para *auto-avaliação*, isto é, não envolverá correção nem feedback pelo professor. Caberá ao aluno demonstrar disciplina e organização de tempo para executá-las, visto que são planejadas para preencher o componente I (4 horas/sem.) previsto no nosso ementário. Convencionamos como critérios de auto-avaliação:

- **Satisfatório:** solucionou todos os problemas/requisitos da atividade formativa;
- **Pouco satisfatório:** solucionou alguns problemas/requisitos da atividade formativa;
- **Insatisfatório:** não tentou fazer a atividade formativa.

Com base em ofertas anteriores, pode-se afirmar que há uma correlação forte e positiva entre desempenho satisfatório nas atividades formativas e notas altas nas provas, conseqüentemente quem efetua as atividades formativas costuma ser aprovado na disciplina. Portanto, caso o aluno sistematicamente não efetue as atividades formativas, deve preocupar-se, pois provavelmente terá desempenho insatisfatório nas provas e no projeto, o que implica em reprovação.

<sup>1</sup>Para uma introdução breve sobre essa metodologia, vide [https://en.wikipedia.org/wiki/Active\\_learning](https://en.wikipedia.org/wiki/Active_learning)

<sup>2</sup>[https://en.wikipedia.org/wiki/Flipped\\_classroom](https://en.wikipedia.org/wiki/Flipped_classroom)

<sup>3</sup>[https://en.wikipedia.org/wiki/Problem-based\\_learning](https://en.wikipedia.org/wiki/Problem-based_learning)

## 8 Critérios de avaliação somativa

A avaliação somativa consiste nos componentes dados pela Equação 1, onde:

$$N_F = 0.3 \cdot N_{p1} + 0.4 \cdot N_{p2} + 0.3 \cdot N_{proj} \quad (1)$$

- $N_{p1}$  é a nota da Prova 1;
- $N_{p2}$  é a nota da Prova 2;
- $N_{proj}$  é a nota do projeto final.

As provas são individuais e consistem na construção de uma aplicação, completa ou parcialmente, com base em um conjunto de requisitos enunciados. A depender da complexidade do tema, é possível que uma prova seja organizada em fases, parte extra-classe e parte em classe — essa dinâmica será propriamente enunciada na ocasião mais adequada para a prova.

O conceito final será obtido de acordo com a Equação 2.

$$C_F = \begin{cases} A, & \text{se } N_F \in [8.5, 10.0] \\ B, & \text{se } N_F \in [7.0, 8.5) \\ C, & \text{se } N_F \in [5.5, 7.0) \\ D, & \text{se } N_F \in [5.0, 5.5) \\ F, & \text{se } N_F \in [0.0, 5.0) \\ O, & \text{se ausência exceder 25\%} \end{cases} \quad (2)$$

## 9 Critérios de avaliação para atividades de programação

Tanto as provas quanto o projeto se qualificam como atividades de programação. No caso das atividades que envolvam prazo de entrega, não serão aceitas as que forem entregues fora do prazo. A nota máxima de cada atividade será obtida apenas se a mesma for entregue no prazo e executada correta e completamente.

A nota do projeto final será calculada de acordo com um conjunto de critérios, que encontra-se publicado em documento específico de enunciado do projeto. Adicionalmente, o projeto deverá ser entregue incrementalmente, de acordo com calendário a ser divulgado naquele mesmo documento, sendo que cada entrega corresponde a uma fração da nota total do projeto.

Os programas solicitados em atividades de avaliação serão analisados quanto aos seguintes critérios gerais:

- **Eficiência:** os programas desenvolvidos deverão ter bom desempenho, o que pode englobar o tratamento adequado dos seguintes fatores:
  - ler e escrever dados nas quantidades mínimas necessárias para resolver o problema;
  - não desperdiçar memória primária (RAM);
  - acessar memória secundária (disco) somente quando necessário;
  - minimizar uso de rede, tanto em tempo de operação quanto em quantidade de dados transmitidos, com atenção aos requisitos de intermitência devido à mobilidade;
  - entre outros.
- **Acurácia:** o programa deverá atender adequadamente a todos os requisitos enunciados para a atividade;
- **Estrutura e organização do código:** atentar principalmente aos seguintes aspectos:
  - **Auto-documentação:** nomes intuitivos para variáveis e métodos/funções;
  - **Modularização:** funções/métodos com alta concisão e baixo acoplamento, isto é, que sejam em sua maioria curtos, e que realizem preferencialmente uma única tarefa;
  - **Comentários:** documentação completa porém ao mesmo tempo concisa (sem poluição visual, apenas nos lugares adequados e necessários). Programar é redigir numa linguagem formal de alto nível, que será interpretada tanto por computadores quanto por humanos.

## 9.1 Mecanismos de avaliação substitutivos

A prova substitutiva será aplicada ao aluno que atender às seguintes condições simultaneamente: *i)* possuir pelo menos 75% de participação; e *ii)* possuir justificativa de ausência em uma das provas. A listagem dos documentos aceitos como justificativa consta na resolução ConsEPE nº 227<sup>4</sup>.

A nota obtida na prova substitutiva necessariamente substituirá a prova para a qual o aluno tem justificativa. Se porventura houver justificativa para ambas P1 e P2, a segunda prova substitutiva será realizada em data a ser negociada com o professor, pois não está prevista no calendário da disciplina.

## 9.2 Mecanismo de recuperação

A recuperação será aplicada apenas aos alunos que tiverem conceito final D ou F. Ocorrerá na data especificada no cronograma, no mesmo horário da aula.

A nota obtida na prova de recuperação ( $N_R$ ) será usada para obter a nota final com recuperação ( $N_{FR}$ ), que consiste na média estabelecida pela Equação 3.

$$N_{FR} = \frac{N_F + N_R}{2} \quad (3)$$

O conceito final obtido na recuperação ( $C_{FR}$ ) é o conceito que entrará no histórico, obtido de acordo com os limiares para a nota final de recuperação ( $N_{FR}$ ) dados pela Equação 4.

$$C_{FR} = \begin{cases} C, & \text{se } N_{FR} \geq 5.5 \\ D, & \text{se } N_{FR} \in (5.0, 5.5) \\ F, & \text{se } N_{FR} \leq 5.0 \end{cases} \quad (4)$$

No caso dos alunos que não participarem da recuperação,  $C_{FR} = C_F$ .

## 10 Código de honra

A aprovação na disciplina é baseada exclusivamente no esforço e trabalho pessoal do discente, ao qual cabe garantir que não ajudará ou receberá ajuda não-permitida em qualquer atividade usada pela equipe docente para fins de avaliação (e.g. provas, trabalhos, listas, etc.).

Exemplos de violação do código de honra incluem:

- Copiar atividades avaliativas (e.g. listas, trabalhos, provas, etc.) ou permitir que outros discentes copiem suas atividades avaliativas;
- Colaboração não-permitida entre indivíduos ou grupos (e.g. oferecer vantagens em troca de soluções prontas, doar trechos para o trabalho de outro grupo, etc.);
- Permitir que outros assumam sua identidade em atividades avaliativas (e.g. entregar trabalho que não fez ou permitir que outros façam provas por você);
- Plágio (i.e. aplicável a textos, programas de computador, etc.);
- Receber ou conceder ajuda em atividades avaliativas quando o contexto mostra que não é sensato receber tal ajuda.

Como consequências de violação do código de honra tem-se:

- Reprovação automática na disciplina, com conceito F;
- Denúncia na Comissão de Transgressões Disciplinares Discentes da Graduação, a qual decidirá sobre a punição adequada à violação, o que pode levar a advertência, suspensão ou desligamento, de acordo com os arts. 78-82 do Regimento Geral da UFABC.

<sup>4</sup><http://prograd.ufabc.edu.br/normas>

## 11 Cronograma de aulas

O plano a seguir pode variar de acordo com o aproveitamento aferido nas turmas durante o quadrimestre. As aulas 23 e 24 correspondem a reposições feriados.

Aula #	Dia	Tema
1	03/06	Apresentação da disciplina
2	05/06	Fundamentos da plataforma e das aplicações Android
3	10/06	Atividades
4	12/06	Componentes de layout e agrupamento
5	17/06	Componentes de interface
6	19/06	Comunicação entre componentes: Intent e IntentFilter
7	24/06	Fragmentos e adaptação de conteúdo
8	26/06	Multitouch, gestos, recursos gráficos e animações
9	01/07	Prova 1
10	03/07	Serviços, notificações e receptores de broadcast
-	08/07	Feriado (Revolução constitucionalista)
11	10/07	Concorrência e Redes I
12	15/07	Concorrência e Redes II
13	17/07	Arquivos e configurações
14	22/07	Armazenamento local I
15	24/07	Armazenamento local II
16	29/07	Arquitetura de aplicações I
17	31/07	Arquitetura de aplicações II
18	05/08	Multimídia
19	07/08	Localização e mapas
20	12/08	Prova 2
21	14/08	Prova substitutiva e acompanhamento de projetos
-	19/08	Feriado (em SBC, recesso em SA)
22	21/08	Apresentação de projetos
23	29/08	Apresentação de projetos <sup>5</sup>
24	02/09	Prova de recuperação <sup>6</sup>

<sup>5</sup>Reposição ref. 08/07

<sup>6</sup>Reposição ref. 29/08