PRAGMATIC THINKING AND LEARNING

TIPS

i. Always consider the context.

ii. Use rules for novices, intuition for experts.

iii. Know what you don't know.

iv. Learn by watching and imitating.

v. Keep practicing in order to remain an expert.

vi. Avoid formal methods if you need creativity, Intuition or inventiveness.

vii. Learn the skill of learning.

viii. Capture all ideas to get more of them.

ix. Learn by synthesis as well as by analysis.

x. Strive for good design; it really works better.

xi. Rewire your brain with belief and constant practice.

xii. Add sensory experience to engage more of your brain.

xiii. Learn with the R-mode; follow with L-mode.

xiv. Use metaphor as the meeting place between L-mode and R-mode.

xv. Cultivate humor to build stronger metaphors.

xvi. Step away from the keyboard to solve hard problems.

xvii. Change your viewpoint to solve the problem.

xviii. Watch the outliers: "rarely" doesn't mean "never".

xix. Be comfortable with uncertainty.

xx. Trust ink over memory; every mental read is a write.

xxi. Hedge your bets with diversity.

xxii. Allow for different bugs in different people.

xxiii. Act like you've evolved; breath, don't hiss.

xxiv. Trust intuition, but verify.

xxv. Create SMART [Specific; Measurable; Achievable; Relevant; Time-Boxed] objectives to reach your goals.

xxvi. Plan your investment in learning deliberately.

xxvii. Discover how you learn best.

xxviii. Form study groups to learn and teach.

xxix. Read deliberately.

xxx. Take notes with both the R-mode and the L-mode.

xxxi. Write on: documenting is more important than documentation.

xxxii. See it, Do it, Teach it.

xxxiii. Play more in order to learn more.

xxxiv. Learn from similarities, unlearn from differences.

xxxv. Explore, invent, and apply in your environment - safely.

xxxvi. See without judging, and then act.

xxxvii. Give yourself permission to fail; it’s the path to success.

xxxviii. Groove your mind for success.

xxxix. Learn to pay attention.

xl. Make thinking time.

xli. Use wiki to manage information and knowledge.

xlii. Establish rules of engagement to manage interruptions.

xliii. Send less email, and you’ll receive less email.

xliv. Choose your own tempo for an email conversation.

xlv. Mask interrupts to maintain focus.

xlvi. Use multiple monitors to avoid context switching.

xlvii. Optimize your personal workflow to maximize context.

xlviii. Grab the wheel. You can’t steer on autopilot.

1. **CHAPTER 2 [p. 37]**
   a. Acredito ser ainda um “novice” nas ferramentas que preciso utilizar para o meu trabalho
      - Meu nível de habilidade me impacta na forma de resolver os problemas do dia a dia, porque sempre que surge algum empecilho ou coisa diferente, mesmo que em um problema muito semelhante já tenha sido resolvido por mim, eu travo e não consigo avançar.
   b. Advanced Beginner - German Language; Competent - Taekwondo; Proficient - Muay thai; Expert - writing.

2. **CHAPTER 4 [p. 79 & 94-95]**
   a. Make more metaphors
      - A figure of speech in which a word or phrase is applied to an object or action to which it is not literally applicable [1];
      - A thing regarded as representative or symbolic of something else, especially something abstract [2].
   b. Thesaurus - a book or electronic resource that lists words in groups of synonyms and related concepts
   c. http://wordnet.princeton.edu
   d. What words can you add to your workplace lexicon?

3. **CHAPTER 5 [p. 106 & 113 & 116 & 120 & 123]**
   - Self-serving bias; exposure effects; hawthorne effect; nominal fallacy.
   - Start and maintain an engineer’s log of notes from design meetings, coding questions and solutions, and so on. Put a mark next to older entries anu time you have to go back and use it.
   - I was born in the Artist generation, and I think this resonates with me, even though I do agree with some of the other generational principles, like the nomad.
   - INFJ - T → Apoiador

- When in conflict, consider basic personality types, generational values, your own biases, others' biases, the context, and the environment. Is it easier to find a solution to the conflict with this additional awareness?

4. CHAPTER 6 [p. 133 & 136 & 141 & 153 & 157]

- My three most important goals -
  - 1 - Learn deeply algorithms
    - ➢ Specific:
    - ➢ Measurable:
    - ➢ Achievable:
    - ➢ Relevant:
    - ➢ Time-Boxed:
  - 2 - Be a blackbelt in Taekwondo
    - ➢ Specific: It is specific.
    - ➢ Measurable: I can measure it. By six to six months I conquer a new colored belt. So far, I am yellow.
    - ➢ Achievable: Yep.
    - ➢ Relevant: It matters to me, and I do have control over it
    - ➢ Time-Boxed: in 2 and a half years
  - 3 - Conquer my first technology job in the computing area
    - ➢ Specific
    - ➢ Measurable:
    - ➢ Achievable
    - ➢ Relevant: I need this to gain more experience and knowledge in the tech area.
    - ➢ Time-Boxed: I gave myself 8 months to achieve it.
- I currently don't have a job, but I think mathematical and visual intelligence.
- Among my favorite hobbies, I am strongest at the kinesthetic kind of intelligence.
- **Mind mapping**
- Listen carefully to the speakers.

5. CHAPTER 7 [p. 162 & 167 & 179]

- Put yourself in the picture. Anthropomorphism.

- Explore and get used to a problem before diving into the facts.
- Make your projects [professional or personal] safe to explore.

6. **CHAPTER 8 [p. 187 & 190 & 211]**
   - Meditate
   - I like to play with the magic cube. I have also tried walking around.
   - The instagram is something that really distracts me.
   - In the afternoon.

O PROGRAMADOR PRAGMÁTICO

DICAS

i. Preocupe-se com o seu trabalho.

ii. Reflita sobre o seu trabalho.

iii. Forneça opções, não dê desculpas esfarrapadas.

iv. Não tolere janelas quebradas.

v. Seja um catalisador de mudança.

vi. Lembre-se do cenário em larga escala

vii. Torne a qualidade parte dos requisitos.

viii. Invista regularmente em sua carteira de conhecimentos.

ix. Analise criticamente o que você vê e ouve.

x. É o que você diz e a maneira como diz.

xi. DRY - Don’t Repeat Yourself.

xii. Facilite a reutilização.

xiii. Elimine efeitos entre elementos não relacionados.

xiv. Não há decisões definitivas.

xv. Use projéteis luminosos para encontrar o alvo.

xvi. Crie protótipos para aprender.

xvii. Programa em um nível próximo ao domínio do problema.

xviii. Estime para evitar surpresas.

xix. Reexamine o cronograma junto ao código.

xx. Mantenha as informações em textos simples.

xxi. Use o poder dos shell de comandos.

xxii. Use um único editor bem.

xxiii. Use sempre o controle do código-fonte.

xxiv. Corrija o problema, esqueça o culpado.

xxv. Não entre em pânico.

xxvi. “Select” não está com defeito.

xxvii. Não suponha, teste.

xxviii. Aprenda uma linguagem de manipulação de texto.

xxix. Escreva um código que crie códigos.

xxx. Você não conseguirá criar um software perfeito.

xxxi. Projete com contratos.

xxxii. Encerre antecipadamente.

xxxiii. Se não pode acontecer, use exceções para assegurar que não aconteça.

xxxiv. Use exceções para problemas excepcionais.

xxxv. Acabe o que começou.

xxxvi. Reduza a vinculação entre módulos.

xxxvii. Configure, não integre.

xxxviii. Coloque as abstrações no código e os detalhes em metadados.

xxxix. Analise o fluxo de trabalho para melhorar a concorrência.

xl. Projete usando serviços.

xli. Projete sempre pensando na concorrência.

xlii. Separe as visualizações dos modelos.

xliii. Use quadros negros para coordenar o fluxo de trabalho

xliv. Não programe por coincidências.

xlv. Estime a ordem de seus algoritmos.

xlvi. Teste suas estimativas.

xlvii. Refatore cedo, refatore sempre.

xlviii. Projete para testar.

xlix. Teste seu software, ou seus usuários testarão.

l. Não use um código de assistente que você não entender.

li. Não colete resquícios, cave-os.

lii. Trabalhe com um usuário para testar como pensa um usuário.

liii. Abstrações têm vida mais longa do que detalhes.

liv. Use um glossário do projeto.

lv. Não pense fora da caixa - encontre a caixa.

lvi. Só comece quando estiver pronto.

lvii. Algumas coisas são fáceis de fazer, mas não de descrever.

lviii. Não seja escravo dos métodos formais.

lix. Ferramentas caras não produzem projetos melhores.

lx. Organize as equipes com base na funcionalidade.

lxi. Não use procedimentos manuais.

lxii. Teste cedo, teste sempre, teste automaticamente.

lxiii. A codificação só estará concluída após todos os testes serem executados.

lxiv. Use sabotadores para testar seus testes.

lxv. Teste a cobertura de estados e não a cobertura do código.

lxvi. Encontre os erros apenas uma vez.

lxvii. Trate o português simplesmente como outra linguagem de programação.

lxviii. Construa a documentação no código, não a acrescente como complemento.

lxix. Exceda gentilmente as expectativas de seus usuários.

lxx. Assine seu trabalho.