Novo Guia de Prompting da OpenAI (GPT-5)

Seção 2 — Como projetar prompts eficazes para o GPT-5

Este guia prático detalha cinco pilares: Responses API, Reasoning Effort, Eagerness/Autonomia, estruturação com XML/Markdown e QA antes da resposta final. Inclui erros comuns, boas práticas e templates prontos para colar.

Como usar este guia

1) Leia cada pilar. 2) Copie os templates para seu app. 3) Ajuste limites e schemas conforme o caso de uso. 4) Use o checklist de QA sempre que a resposta envolver cálculo, inferência ou busca.

1) Responses API (mantém contexto, suporta multimodal)

O que é

Interface mais recente da OpenAI para conversas com estado. Cada resposta pode chamar ferramentas (ex.: web search) e o raciocínio/estado persiste entre chamadas. Aceita entradas de texto e imagem; a saída pode ser texto ou JSON estruturado.

Erros comuns

• Tratar cada pergunta como conversa isolada (o agente repete buscas e esquece resultados anteriores).

• Não definir formato de saída, obrigando parsing frágil de texto.

Faça certo

• Mantenha um fio de estado entre passos e peça saída em JSON.

• Defina objetivo, ferramentas permitidas e critério de parada.

Template de prompt — estado + formato

```
Objetivo: avaliar suplemento proteico a partir do rótulo (imagem anexa).
Ferramentas: web_search (apenas se faltar dado do rótulo).
Fluxo:
1) Extrair dados do rótulo (por imagem).
2) Se faltar fonte proteica ou DV, usar web_search 1 vez.
3) Calcular PDCAAS e resumir.
Saída (JSON):
{ "quality": string, "pdcaas": number, "key_findings": string[], "recommendation": string }
Pare após preencher o JSON.
```

2) Reasoning Effort (low, medium, high)

O que é

Um controle de profundidade de raciocínio do modelo:

• low: rápido e barato, bom para tarefas objetivas.

• medium: equilíbrio entre profundidade e custo.

• high: explora múltiplos caminhos, faz cálculos/checagens extras; mais lento/caro.

Erros comuns

• Usar high para tudo (respostas lentas e caras, às vezes divagação).

• Usar low em tarefas que exigem cálculo/checagem (erros silenciosos ou omissões).

Faça certo

• Combine o esforço ao tipo de tarefa e imponha limites (tempo, número de buscas, chamadas de ferramenta).

Exemplos práticos

```
Exemplo enxuto:
  Pergunta: Some 2+2 e devolva apenas o número.
  Config: Effort=low, Verbosity=low.

Exemplo profundo com limites:
  Tarefa: comparar 5 suplementos e calcular PDCAAS.
  Config: Effort=high; limite máx de 2 buscas web; tempo alvo < 30s.
  Exija: mostrar o cálculo e citar 2 fontes; parar após recomendação.
```

3) Eagerness/Autonomia (agir sozinho vs. seguir passo a passo)

O que é

Grau de autonomia do agente: continuar sozinho até concluir ou devolver a conversa pedindo confirmação. Útil para evitar perguntas em excesso ou cascatas de chamadas sem foco.

Erros comuns

• Autonomia demais: o agente encadeia ferramentas sem objetivo claro.

• Autonomia de menos: o agente pede confirmação a cada passo e não progride.

Faça certo

• Especifique persistência, orçamento de ferramentas e critério de parada/escape sob incerteza.

Templates

```
Autonomia com trilhos:
Você é um agente. Continue até concluir a análise PDCAAS.
Orçamentos: máx 2 web_search; máx 1 reprocessamento de imagem.
Critério de parada: quando houver cálculo, comparação com 2 alternativas e recomendação.
Se faltar dado essencial, assuma valor conservador e declare a suposição no resultado.

Modo cauteloso:
Atue de forma conservadora: não use web_search a menos que a extração do rótulo falhe.
Se houver ambiguidade, faça 1 pergunta de esclarecimento e então conclua.
```

4) Estruturação com blocos XML/Markdown (goal, method, stop_criteria)

O que é

Fornecer uma grade clara com objetivo, método, limites e formato de saída. Reduz dispersão, melhora reprodutibilidade e avaliação.

Erros comuns

• Prompt em parágrafo solto sem prioridades, método ou saída definida.

• Tags inconsistentes ou listas sem ordem.

Faça certo

• Use blocos curtos, listas numeradas e um schema de saída. Inclua critérios de parada e escape sob incerteza.

Template XML/Markdown

```
<goal>
Analisar rótulo, calcular PDCAAS e recomendar uso (atleta, idoso, uso diário)
</goal>

<method>
1) Extrair valores do rótulo (imagem).
2) Se faltar DV ou fontes, 1 busca web.
3) Calcular PDCAAS com trabalho matemático visível.
4) Comparar com 2 alternativas (whey isolado e soja).
</method>

<stop_criteria>
Parar quando houver: cálculo, 2 comparações, recomendação final.
Se houver incerteza, declarar suposições e impacto.
</stop_criteria>

<output_schema>
JSON: { "quality": string, "pdcaas": number, "key_findings": string[], "assumptions": string
</output_schema>
```

5) QA (Quality Assurance) antes da resposta final

O que é

Etapa de checagem final: validação matemática, revisão de fontes, coerência do escopo e clareza da recomendação. Evita erros sutis (ex.: confundir PDCAAS com média simples, ignorar proteínas complementares).

Erros comuns

• Entregar resultado assim que surge “alguma resposta”.

• Não mostrar cálculo ou não citar fontes.

• Não declarar suposições/limitações.

Faça certo

• Exigir bloco de QA com checklist explícito e autoavaliação curta. Exigir cálculo visível, 2 fontes e nota de confiança.

Exemplo de saída com QA

```
Saída (JSON):
{
  "quality": "Boa",
  "pdcaas": 0.84,
  "key_findings": [
    "Fonte principal: whey; colágeno presente reduz qualidade média",
    "Cálculo reproduz DV do rótulo"
  ],
  "assumptions": [
    "Proporção de colágeno inferida por DV e ingredientes",
```

```
    "Sem dados laboratoriais, uso de valores de referência"
  ],
  "recommendation": "Adequado para uso diário; prefira whey isolado para hipertrofia máxima"
}

QA:
- Cálculo conferido: sim (mostrar passo a passo).
- Proteínas complementares consideradas: sim (se mistura vegetal).
- Fontes citadas: 2 referências confiáveis (ex.: FDA, revisão PDCAAS).
- Nota de confiança: 0.78 (0–1); motivo: proporções inferidas.
```

Checklist de QA (cole no fim do seu prompt)

```
Antes de responder:
[ ] Mostrar cálculo matemático (não apenas o resultado)
[ ] Citar 2 fontes confiáveis
[ ] Declarar suposições e impacto
[ ] Checar complementaridade proteica (se mistura vegetal)
[ ] Produzir saída no schema solicitado
```

Apêndice — Templates rápidos

Template: tarefa simples (baixo custo)

```
Config:
- Reasoning Effort: low
- Verbosity: low

Prompt:
Some 2+2 e devolva apenas o número.

Saída esperada:
4
```

Template: investigação com limites (alto esforço controlado)

```
Config:
- Reasoning Effort: high
- Verbosity: medium
- Máx web_search: 2
- Tempo alvo: < 30s

Prompt:
Compare 5 suplementos proteicos. Extraia (ou pesquise) fontes, calcule PDCAAS mostrando o cá
cite 2 fontes de referência e recomende o melhor para idosos e para atletas.

Saída (JSON):
{
  "comparisons": [ { "name": "...","pdcaas": 0.93,"pros": [],"cons": [] } ],
  "recommendations": { "idosos": "...", "atletas": "..." },
  "notes": ["cálculo e suposições visíveis"]
}
```

Template: autonomia com trilhos

```
Você é um agente. Continue até concluir a análise PDCAAS.
Orçamentos: máx 2 web_search; máx 1 reprocessamento de imagem.
Critério de parada: quando houver cálculo, 2 comparações e recomendação final.
Se faltar dado essencial, assuma valor conservador e declare a suposição.
```