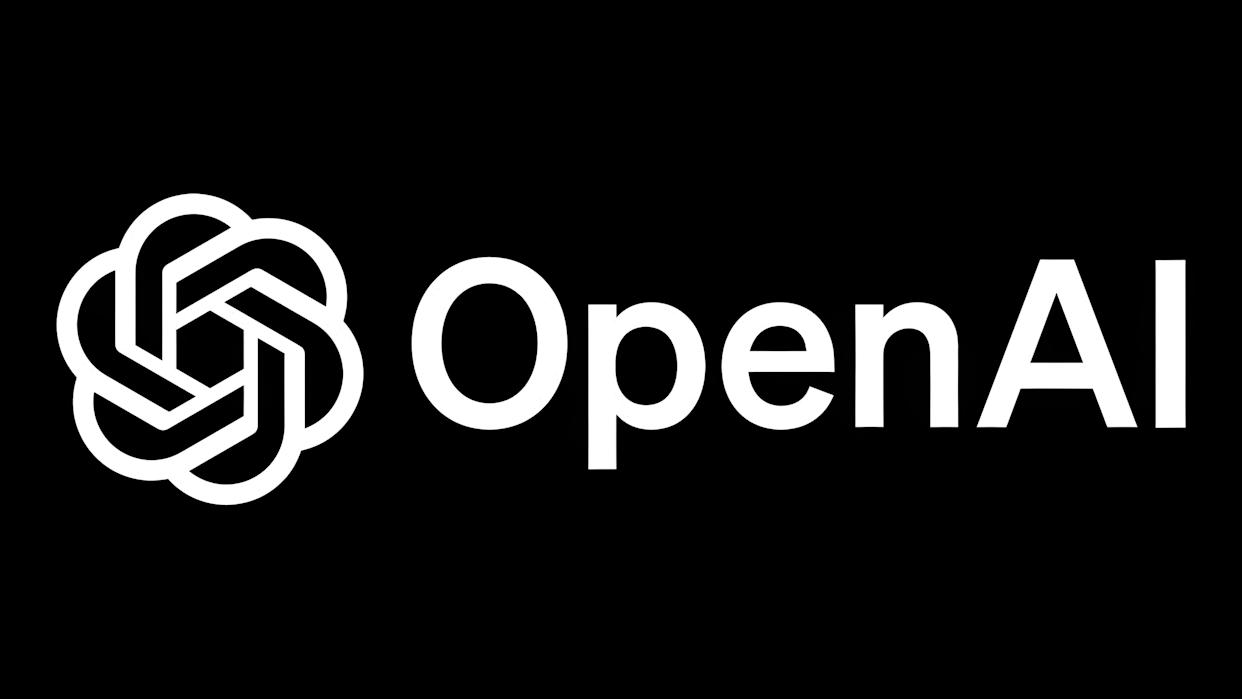
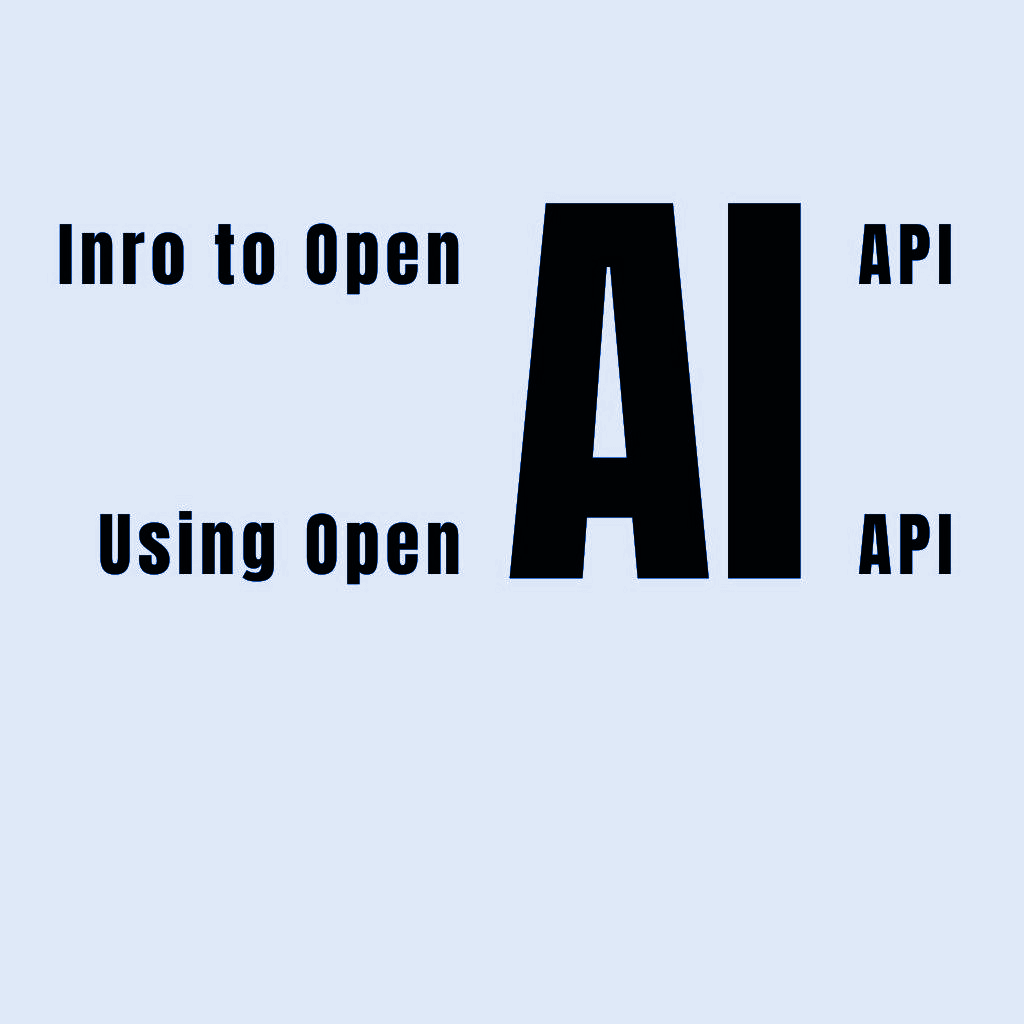
Tab 1

[22, 23, 24, 25, 26, 27, 28]





COURSES

* 22 - [Intro to OpenAI API | Codecademy](https://www.codecademy.com/learn/intro-to-open-ai-gpt-api)
* 23 - [Using OpenAI APIs: Accessing OpenAI APIs from Python | Codecademy](https://www.codecademy.com/learn/ext-courses/using-openai-apis-accessing-openai-apis-from-python)
* 24 - [OpenAI API Coding with Python | Codecademy](https://www.codecademy.com/learn/open-ai-api-coding-with-python)
* 25 - [Proven Techniques for Technical Communication | Codecademy](https://www.codecademy.com/learn/ext-courses/proven-techniques-for-technical-communication)
* 26 - [Writing with a Professional Mindset | Codecademy](https://www.codecademy.com/learn/ext-courses/writing-with-a-professional-mindset)
* 27 - [Telling a Business Story | Codecademy](https://www.codecademy.com/learn/ext-courses/telling-a-business-story)
* 28 - [Agile Principles and Methodologies | Codecademy](https://www.codecademy.com/learn/ext-courses/agile-principles-and-methodologies)
* 29 - [Data Analyst Interview Preparation | Codecademy](https://www.codecademy.com/learn/paths/intprep-data-analyst-interview-prep)
* 30 - [Data Scientist Interview Preparation | Codecademy](https://www.codecademy.com/learn/paths/intprep-ds-ds-interview-prep)

# INDEX

[**Introduction 6**](#_l1jygv3vm5y)

[Endpoints 7](#_c4q6zmx6vibx)

[chat/completions 7](#_8bu991nbxn5b)

[Other Endpoints 7](#_nnvgfj5n4oxr)

[Models, Tokens and Cost 9](#_ocujtvd8n1il)

[Tokens 10](#_ks6nqsfjeibr)

[Cost Analysis 10](#_so5reidpt548)

[Input Prompt 11](#_d59io6m2kcs8)

[Message Roles 11](#_rnqpcnxk7uex)

[Prompt Engineering 11](#_fxqj6ff6ztub)

[**Input Prompt Object 12**](#_lmdia3nc8jk7)

[Token Sampling 12](#_ttfjmnpc6t8r)

[temperature 13](#_xasj0hsdvxx9)

[top\_p 13](#_kobftea74nw)

[Token Repetition 14](#_7i4i7la6lv7f)

[presence\_penalty 14](#_8t2cghs7022a)

[frequency\_penalty 14](#_b288kqa7buh3)

[Text Completion Response 15](#_v6qud2btbwrr)

[choices 15](#_iothiodwk8k)

[Text Completion Response 16](#_ke4ue5ksti1j)

[choices 17](#_qw0tzrpfle6a)

[**Response Object 18**](#_eb2evudi4n97)

[Summary 19](#_blqelbw7kz3y)

[**Next Steps 19**](#_j1pxtepfagrt)

[Intro to Language Models in Python 20](#_75kznufos5vy)

[Intro to ChatGPT 20](#_c9e9cb1n6qys)

[Unit Testing with Generative AI Case Study 20](#_6xhak3weowdw)

[Pair Programming with ChatGPT Case Study 20](#_clr4nn6wpeb)

[**Using OpenAI APIs: Accessing OpenAI APIs from Python 21**](#_hc410vacqh9)

[Table of Contents 22](#_lsc5wy1aqhuc)

[1. Video: Course Overview (it\_aiopapdj\_02\_enus\_01) 23](#_abtw5rxnxzbr)

[2. Video: Generating Application Programming Interface (API) Keys to Use with Endpoints (it\_aiopapdj\_02\_enus\_02) 24](#_3sqj4hourixz)

[3. Video: Making API Requests with cURL (it\_aiopapdj\_02\_enus\_03) 29](#_frjn8uk0lu18)

[4. Video: Making Our First API Request Using Python (it\_aiopapdj\_02\_enus\_04) 36](#_lryo8jdde1m4)

[5. Video: Exploring Chat Completions and JSON Mode (it\_aiopapdj\_02\_enus\_05) 43](#_7c2va1165b5q)

[6. Video: Streaming Chat Completions with the Legacy Completions API (it\_aiopapdj\_02\_enus\_06) 51](#_3pl2vtv0gxfa)

[7. Video: Exploring Chat and Completion Models (it\_aiopapdj\_02\_enus\_07) 56](#_y4elj3ixt9ml)

[8. Video: Setting up Utility Functions to Explore Model Determinism (it\_aiopapdj\_02\_enus\_08) 62](#_r41jumm1gwj3)

[9. Video: Using the Seed Parameter for Almost Deterministic Responses (it\_aiopapdj\_02\_enus\_09) 70](#_nqwpsnthsg3u)

[10. Video: Tweaking Model Parameters – Temperature and Max Tokens (it\_aiopapdj\_02\_enus\_10) 78](#_tfeil41xthp4)

[11. Video: Tweaking Model Parameters – Stop Sequences Top P and Penalties (it\_aiopapdj\_02\_enus\_11) 84](#_qk0yiwnuxhgz)

[12. Video: Course Summary (it\_aiopapdj\_02\_enus\_12) 90](#_2at0oik6rdn9)

[Course File-based Resources 92](#_buj0o0gn12nv)

[Introduction to OpenAI API 93](#_5h1cfw8g048b)

[Generative AI 94](#_271ju7shzrnt)

[ChatGPT and OpenAI API 94](#_gnphj53e5vo9)

[OpenAI Module 95](#_uxkh4gk0uarw)

[OpenAI Class 95](#_27318aicd07)

[Models 95](#_onysmslke0d3)

[OpenAI API Authentication 96](#_up5903m9h8t2)

[First Completion 96](#_wnprwjl05rjb)

[Using a System Prompt 99](#_ig629a8eqenl)

[Prompt Engineering 99](#_73lpo1nuv2dp)

[Objectives 104](#_o664ddria8gj)

[Job Aid: Techniques for Clear and Effective Technical Communications 104](#_rchfnc65n0r4)

[**Proven Techniques for Technical Communication 106**](#_omxdeb7o37xe)

[Table of Contents 106](#_f5pu8pj4d3oi)

[1. Video: Proven Techniques for Technical Communication (bs\_acm22\_a01\_enus\_01) 106](#_2sy2iqi96mnm)

[2. Video: Considerations for Creating Technical Communications (bs\_acm22\_a01\_enus\_02) 107](#_6181g6e66izp)

[3. Video: Know Your Audience (bs\_acm22\_a01\_enus\_03) 108](#_gr2j25ox9oyv)

[4. Video: Acquiring Subject Matter Knowledge (bs\_acm22\_a01\_enus\_04) 110](#_1aux2ew7qrrf)

[5. Video: Creating Clear and Effective Technical Communications (bs\_acm22\_a01\_enus\_05) 111](#_j48wv4ybcwtc)

[6. Video: Using Visual and Graphic Aids to Enhance a Technical Communication (bs\_acm22\_a01\_enus\_06) 112](#_qzvnrfsttabj)

[7. Knowledge Check: Creating Technical Communications 113](#_l1uvkfk0wqgo)

[Question 1: Matching 113](#_ub5n85hat3xe)

[Options: 113](#_86yckvtjqfuc)

[Targets: 113](#_4ssinz5diqr5)

[Answer 114](#_jqz394sttpx5)

[Feedback: 114](#_7odqkk133nj5)

[Question 2: Matching 114](#_afxxvtlqf45r)

[Options: 114](#_cslk55lfssat)

[Targets: 114](#_agxbgf7v3qpk)

[Answer 114](#_9sat5qmvgwlp)

[Feedback: 115](#_o09jb1448vey)

[Question 3: Multiple Choice 115](#_n085uvhmis4c)

[Options: 115](#_h3d3bnpw4mub)

[Answer 115](#_h7for3wy6o1d)

[Feedback: 115](#_5nd1571uxgib)

[Question 4: Matching 116](#_nd4jlrclh2w6)

[Options: 116](#_9p0l51xacdm2)

[Targets: 116](#_rp5qlbwouriz)

[Answer 116](#_b9nhe72rvtte)

[Feedback: 117](#_a22bp4g0ozsw)

[Question 5: Matching 117](#_aek7nnixghwo)

[Options: 117](#_caow14wbd99r)

[Targets: 117](#_4wxnhdvpw0fv)

[Answer 118](#_e3874ggl51wj)

[Feedback: 118](#_dai3unwo21g1)

[Course HTML Resources 118](#_mn9aclbzz3m6)

[Objectives 120](#_sqjn4e2y614w)

[**● discover the key concepts covered in this course 120**](#_icge9qojh1i6)

[**● identify the essential aspects necessary for crafting professional communication 120**](#_icge9qojh1i6)

[**● recognize strategies for crafting effective and professional written communication 120**](#_icge9qojh1i6)

[**● recognize how various channels of communication can be optimized for impactful professional writing 120**](#_icge9qojh1i6)

[**● recognize best practices for leveling up your professional writing skills 120**](#_icge9qojh1i6)

[**● reflect on what you've learned 120**](#_ma8m62h9p432)

[**Writing with a Professional Mindset 121**](#_hxn00jy0vs38)

[Table of Contents 121](#_yx075p11lmg9)

[1. Video: Writing with a Professional Mindset (bs\_acm36\_a01\_enus\_01) 121](#_sdfs76bz9cn)

[2. Video: Mastering the Essentials of Professional Writing (bs\_acm36\_a01\_enus\_02) 122](#_1u6h29pivhw)

[3. Video: Crafting Impactful Professional Communication (bs\_acm36\_a01\_enus\_03) 124](#_uzu3jhxv5pwx)

[4. Knowledge Check: Developing Professional Writing Skills 126](#_wu7u9zmw3m6s)

[Question 1: Multiple Choice 126](#_xn3v7mjnh2gy)

[Options: 126](#_pz1ship4bza4)

[Answer 126](#_4dqnqbyhntl2)

[Feedback: 127](#_x98koqasbwr1)

[Question 2: Multiple Choice 127](#_5bfm7q44jiqs)

[Options: 127](#_n57p6737rc6x)

[Answer 127](#_vlf6qokx0dde)

[Feedback: 128](#_ajnhcbjngj2d)

[5. Video: Tailoring Your Communication for Different Channels (bs\_acm36\_a01\_enus\_04) 128](#_aa6yplh2whv3)

[6. Video: Leveraging Best Practices to Improve Writing Skills (bs\_acm36\_a01\_enus\_05) 131](#_kb1h4h9uonfs)

[7. Knowledge Check: Enhancing Professional Writing Skills 133](#_mvbm0dwh9hvx)

[Question 1: Matching 133](#_yp1w6i2twl5h)

[Options: 133](#_vld22knrh3g)

[Targets: 133](#_tz04cq8rzchc)

[Answer 133](#_loj9hlawh0ki)

[Feedback: 133](#_ckli62fajoaf)

[Question 2: Multiple Choice 134](#_crow62kgw3fv)

[Options: 134](#_s5w69iby9jnq)

[Answer 134](#_cejxpyux4w7t)

[Feedback: 134](#_s1kttrzeajoj)

[8. Video: Let's Review (bs\_acm36\_a01\_enus\_06) 135](#_lfib6ldokrf4)

[Course HTML Resources 136](#_fz7to1he84nf)

[Duration 140](#_92uhfsnxbdy5)

[Prerequisites 140](#_ybfmi39bp940)

[Expertise Level 140](#_5hfjmqjaq6y1)

[Code 141](#_n49ccbngc4oz)

[Training Credits 141](#_nzf09e6t60kz)

[Objectives 141](#_f7c41of5ncvt)

[**Agile Principles and Methodologies 141**](#_n5ntposrqsz7)

[Table of Contents 141](#_gh2wmmir0jaj)

[1. Video: Agile Principles and Methodologies (bs\_apj13\_a01\_enus\_11) 142](#_ptm86ywijiai)

[2. Video: Understanding Agile (bs\_apj13\_a01\_enus\_01) 142](#_tbtpaw5vj2s)

[3. Video: Agile Values and Principles (bs\_apj13\_a01\_enus\_02) 144](#_hcsl2fbb4caw)

[4. Video: Agile Project Management Model (bs\_apj13\_a01\_enus\_03) 146](#_w5hjz7opwdgq)

[5. Video: Agile Methodologies (bs\_apj13\_a01\_enus\_04) 147](#_3t3g3vt4gmvi)

[6. Video: Scrum Framework (bs\_apj13\_a01\_enus\_05) 149](#_rjodgokyif1d)

[7. Video: Adopting an Agile Approach (bs\_apj13\_a01\_enus\_06) 150](#_e0pg2b4m3uhk)

[8. Video: Initiating an Agile Project (bs\_apj13\_a01\_enus\_07) 151](#_jnenpebe5kbm)

[9. Video: Creating Vision and Charting a Project (bs\_apj13\_a01\_enus\_08) 153](#_ts9qskhghybr)

[10. Video: Agile Contracts (bs\_apj13\_a01\_enus\_09) 154](#_umhxtmoc9bq4)

[11. Video: Agile Documentation (bs\_apj13\_a01\_enus\_10) 155](#_njgb9qqxinb7)

[12. Knowledge Check: Key Agile Concepts 157](#_jefwaggootqo)

[Question 1: Multiple Choice 157](#_ti581mw6kzr6)

[Options: 157](#_6areen4bc5tt)

[Answer 157](#_56ofc61xcvok)

[Feedback: 157](#_hf1uqsiw8l2j)

[Question 2: Multiple Choice 158](#_kun3des7h091)

[Options: 158](#_ksf1x5n5e6cq)

[Answer 158](#_v2ow9puweghk)

[Feedback: 158](#_y3w66d1y3trf)

[Question 3: Multiple Choice 159](#_cjaryo62vr5e)

[Options: 159](#_68w8h9nuzaox)

[Answer 159](#_rlnvn5m2oiwv)

[Feedback: 160](#_upu0xh7c3y9q)

[Question 4: Multiple Choice 160](#_7qe7hr1q945i)

[Options: 160](#_u7vpeblykrnm)

[Answer 160](#_y3s3k3gn3yom)

[Feedback: 161](#_9t2p9x6aydlh)

[Course HTML Resources 161](#_2br74ajcz5ki)

* [**https://www.codecademy.com/courses/intro-to-open-ai-gpt-api/lessons/open-ai-api/exercises/introduction**](https://www.codecademy.com/courses/intro-to-open-ai-gpt-api/lessons/open-ai-api/exercises/introduction)

22 Intro to Open AI

# Introduction

**OpenAI API**

An Application Programming Interface (API) is an excellent way for developers to interact with third-party software easily. Data and functionality can be accessed by sending requests to different API endpoints. Using APIs is a great way to expand your project’s scope and capabilities.

The OpenAI API is no different because it is a portal into the world of large language models and brings generative AI into our

[code](https://www.codecademy.com/resources/docs/ai/generative-ai/code)

Preview: Docs Loading link description

. Understanding this API is important when working with generative AI in your code and projects.

Generative AI applications like

[ChatGPT](https://www.codecademy.com/resources/docs/ai/chatgpt)

Preview: Docs Loading link description

use large language models (LLMs). LLMs, in turn, are built using a technology known as Generative Pre-trained Transformer (and they are named after it, hence GPT-4). A main driver of these models is the Neural Network. Based loosely on the human brain’s functionality,

[neural networks](https://www.codecademy.com/resources/docs/ai/neural-networks)

Preview: Docs Loading link description

are trained on a large amount of data to reproduce certain behaviors.

While powerful and very effective, neural networks operate as a black box because their inner workings are a mystery to even the most highly-trained data scientists. This results in unverifiable model output, where the input and outputs are always processed with a certain amount of randomness.

This lesson will examine the OpenAI API and explain how to create more reliable, controlled outputs from LLMs. We will explore the API’s different generative AI tasks and model offerings, examine the API request and response objects, and look at effectively creating input prompts to receive the desired

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

output. We will also look at controlling the output with hyperparameters to counteract the model’s non-deterministic or random behavior.

**Instructions**

Try out the AI Chat interface by providing prompts and observing what replies you get from the model.  
Use the prompt field that reads “Send a message” to enter your first prompt. Click the “Submit prompt” button at the right end of the field when you are done crafting the prompt.  
The first step to understanding an app like this is learning how the OpenAI API works and the resources it uses to produce accurate and pertinent responses.

### **Endpoints**

The OpenAI API can be used for

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

completion, image generation and modification, audio transcription, and more. Text completion, including

[code](https://www.codecademy.com/resources/docs/ai/generative-ai/code)

Preview: Docs Loading link description

generation, is a popular use case for large language models and is what we will focus on in this course.

API endpoints are server locations where data can be sent and received using code or other means. OpenAI’s API primary text completion endpoint is chat/completions.

###### **chat/completions**

The chat/completions endpoint generates text from a single prompt or defines a multi-prompt chat.

The chat functionality allows us first to set the “behavior” of the chat replies, such as “You are to respond to 8th-grade social studies questions.” Once this is defined, the responses will be on an 8th-grade social studies class level, and questions not involving social studies will potentially be rejected.

Using this endpoint, we can also specify a small number of prompt/reply examples to help guide future responses. This is known as *few-shot prompting* and can profoundly affect the chat’s behavior.

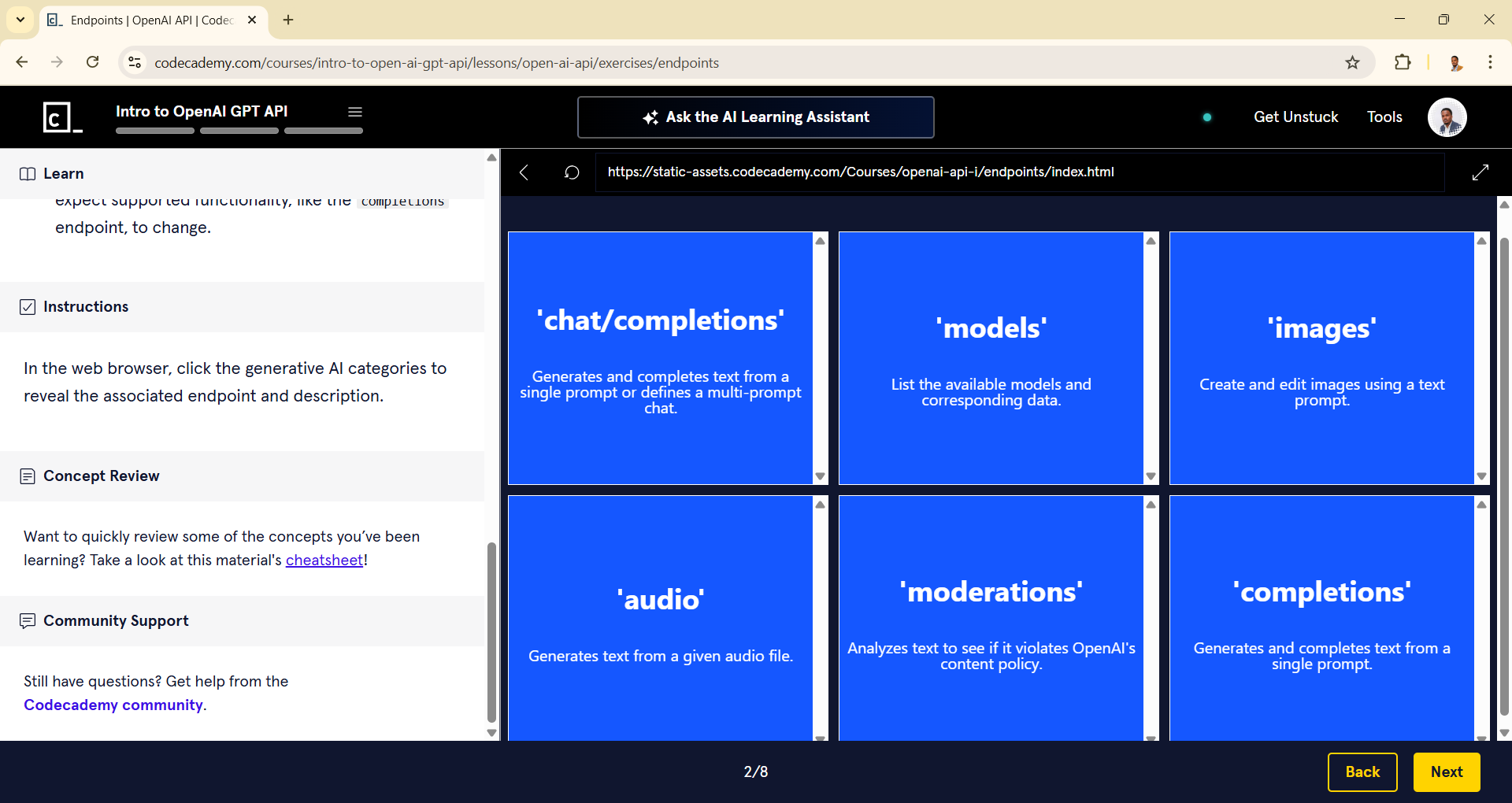
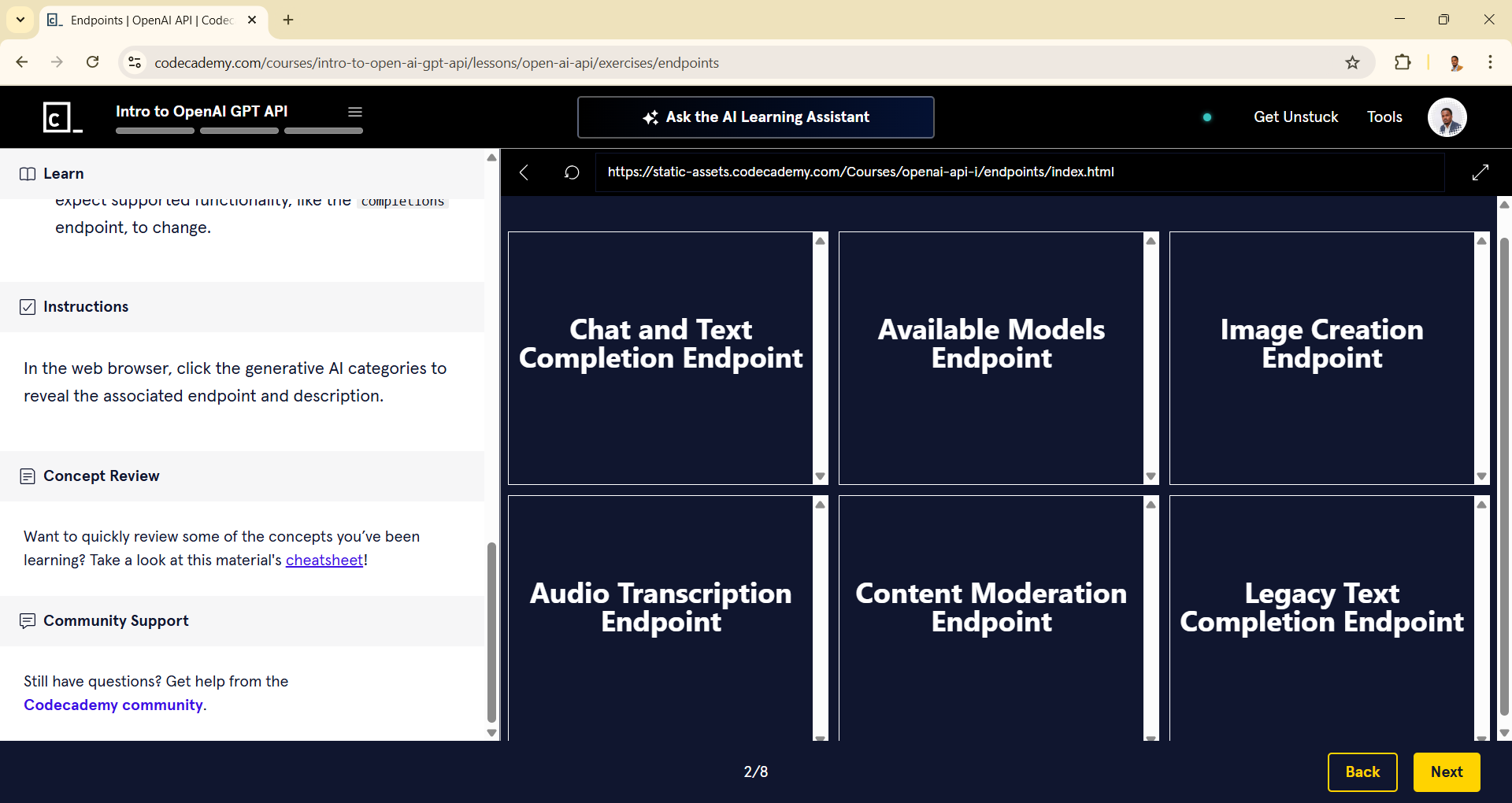
###### **Other Endpoints**

While we will be focusing on the chat/completions endpoint in this course, it is good to know about some of the other endpoints accessible with the API:

* **models**: List the available models and corresponding data such as owner and permissions
* [**images**](https://www.codecademy.com/resources/docs/ai/generative-ai/images)
* **Preview: Docs GenAI can generate new images from existing text prompts and images. Because of a random seed, the images generated are unique creations. GenAI uses diffusion models to create these new unique images.**
* : Create and edit images using a text prompt
* **audio**: Generate text from a given audio file
* **moderations**: Analyze text to see if it violates OpenAI’s content policy, which is useful when passing a user’s input through to the API
* **completions** (Legacy): an older endpoint for producing a text completion using a single input prompt. The AI landscape is changing rapidly, so expect supported functionality, like the completions endpoint, to change.

**Instructions**

In the web browser, click the generative AI categories to reveal the associated endpoint and description.



### **Models, Tokens and Cost**

There are many different LLMs available through the OpenAI API. Each model has varying performance and capabilities and can only be accessed through certain API endpoints. For the chat/completions endpoint, various *GPT-3.5* and *GPT-4* models are available.

There are many differences between the available models, but one important measurement is the number of parameters in a model. Not to be confused with hyperparameters, a model’s parameters are the tuned values used by the neural nets in the large language model that allow the model to process the input and generate the output. If you like to think of things in terms of math, you can think of these parameters as weighted coefficients.

OpenAI has hinted that GPT-4 has over 1 trillion parameters, which makes it way bigger than GPT-3.5’s 175 billion parameters.

###### **Tokens**

*Tokens* are words, or pieces of words, and are what LLMs use to parse the given input and generate an output. Each model has a maximum number of tokens it can process, and the cost of using each model is based on the number of tokens in the input prompt and the generated output.

A general rule is that about **750 words equals 1000 tokens**. When picking what OpenAI model to use, the cost is given in an amount per 1 million tokens. The most cost-effective model in the chat/completions endpoint is gpt-3.5-turbo.

###### **Cost Analysis**

A less expensive model, such as GPT-3.5-turbo, might require more generations of

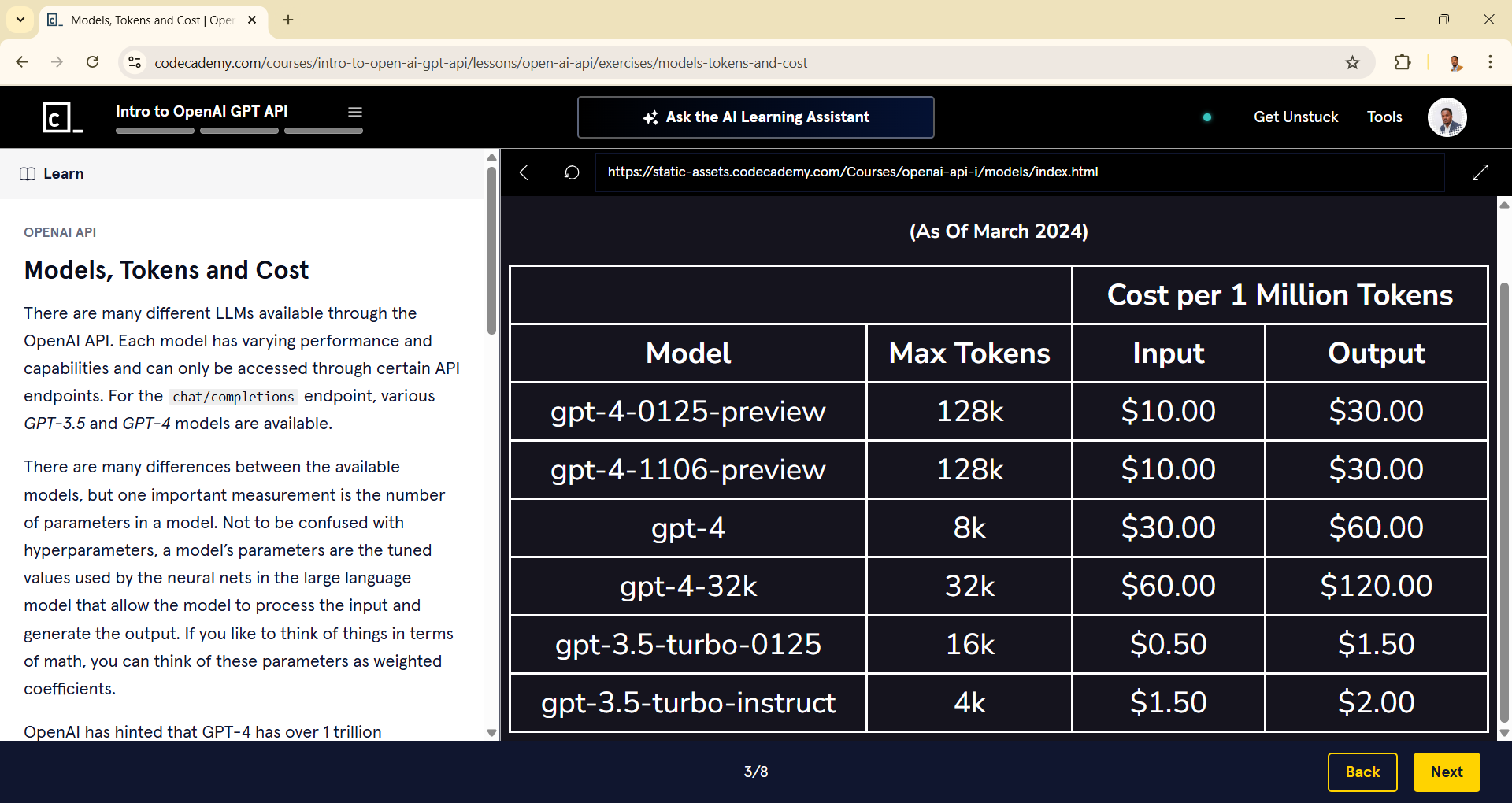
[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Generative AI can create blogs, write ad copy, create new content based on text input, and is capable of summarizing and changing the style and tone of text content.

completions to get close to a desired output. Sometimes, using the costly model will require fewer generations to reach that final product. It might also be the case that using any GenAI at all is more expensive than hiring people to accomplish the same task. This cost analysis is essential to choose the best path forward.

**Instructions**

1. The table in the web browser details several common text completion models, maximum tokens, and costs. There are a few things to note regarding this information:
   * OpenAI is always updating the costs of their models so these values may change at any time.
   * gpt-3.5-turbo is by far the most cost-efficient model. While it does not have as many parameters as gpt-4, it is considered the best price-to-performance.



### **Input Prompt**

LLMs are known for generating

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

, but it is essential to remember that they also have to parse the input prompt text before they generate the output. The chat/completions endpoint allows us to send input messages with different roles. This will enable us to supply multiple inputs that represent a conversation.

###### **Message Roles**

The chat/completions endpoint requires information on the input’s role and content, where role defines the purpose of the given content. A few of the roles are:

* system: Used once to define the chatbot’s behavior. This can guide the output toward specific topics.
* user: This role can be used independently as a single prompt. It can also be used in *few-shot prompting*: where a user prompt is paired with an assistant prompt to give chat examples and define how the chat should respond in the future.
* assistant: This is the role of the LLM and is assigned to the generated reply. As mentioned above, it can also be submitted as an input with the user role to give an example reply. This is useful for few-shot prompting.

###### **Prompt Engineering**

A well-designed input prompt must be descriptive while efficiently using tokens, whether creating a single input prompt with either endpoint or using few-shot prompting with the chat endpoint. Crafting input prompts to get the desired output is known as *prompt engineering*. The following are a few strategies to create good prompts:

* **Be Descriptive**: Use adjectives and descriptive language to give the model more context on what to output.
* **Be Specific**: Avoid generalizing with words like “a few” and instead use “three”.
* **Define the Output**: Ask for the output to be in a certain format, such as JSON or listed out.
* **Provide an Example**: If you have an example of the end result you’re looking for, include it in your prompt.

To provide input to the chat/completions endpoint, the data is structured in a way that the model can interpret and process effectively. Two key fields are essential in this object: "model" and "messages". The "model" field specifies the language model to be used for generating text completions, such as "gpt-3.5-turbo". The "messages" field is a group of role-content pairs representing the conversation’s flow.

| Input Prompt Object {  "model": "gpt-3.5-turbo",  "messages": [  {  "role": "system",  "content": "Phrase all answers based on a 5th grade reading level."  },  {  "role": "user",  "content": "What is a perimeter?"  }  ]  } |
| --- |

### **Token Sampling**

[Large Language Models (LLMs)](https://www.codecademy.com/resources/docs/ai/foundation-models/large-language-models)

Preview: Docs Large Language Models are artificial intelligence systems that are designed to process and generate human language on a massive scale.

are built by training

[neural networks](https://www.codecademy.com/resources/docs/ai/neural-networks)

Preview: Docs A neural network is a method in artificial intelligence that teaches computers to process data in a way that is inspired by the human brain.

on large volumes of

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

to learn the probability distributions of the next best word (or token) to choose in a given context. They can use these probabilities to select and combine tokens to create readable output text. A deterministic (less-random) output would be a selection of the higher probability tokens. This text might be very close to the text used in the LLMs training data. Adjusting how these probabilities are sampled can affect the determinism of the completion output.

There are different use cases when adjusting the determinism of your output. You might not have a large sample of lower probability tokens for

[code](https://www.codecademy.com/resources/docs/ai/generative-ai/code)

Preview: Docs Loading link description

completion since there may only be a single solution for a code snippet. In contrast, a more creative task (like requesting a story including quick character and location descriptions) could result in many token probabilities to sample from. A less deterministic output might result in something more random and unexpected.

The API hyperparameters controlling token probability sampling are temperature and top\_p.

##### **temperature**

The temperature hyperparameter adjusts how the token probabilities are calculated. The range of this hyperparameter is from 0 up to 2, where a lower temperature is more deterministic, and a higher temperature is less deterministic (or more random).

A lower temperature will squish a probability distribution to have a narrower and taller distribution curve, resulting in a few token probabilities with very high probabilities. This means that outcomes that were already more likely get higher priority, keeping the LLM outputs closer to its training text.

A higher temperature results in flattening a distribution curve, resulting in more token probabilities that are closer in value. This also means that previously less likely outcomes now have a greater chance of being picked!

##### **top\_p**

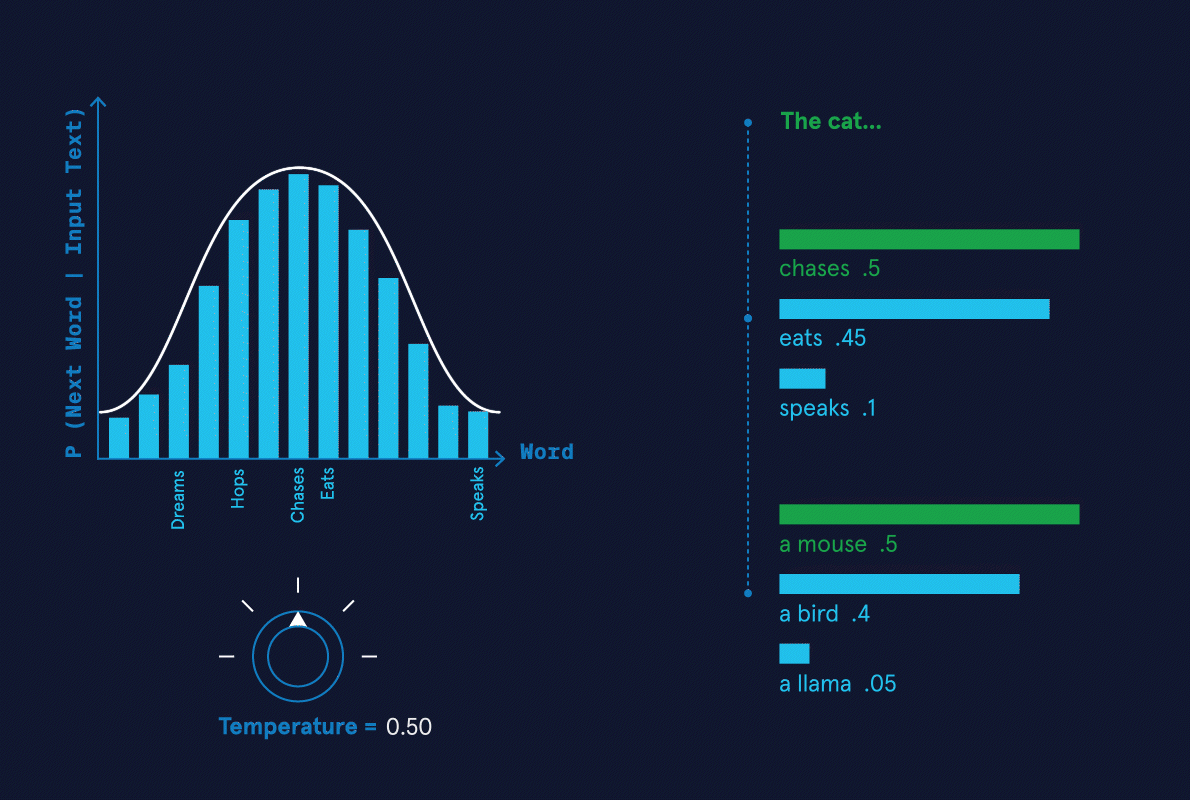
The top\_p hyperparameter is short for top percentage. The range of this hyperparameter is from 0 up to 2, and it adjusts which portion of the highest token probabilities are sampled. By choosing a certain percentage of the top token probabilities, we directly affect how probable each token is.

For example, look at the following probability distribution of 4 potential tokens:

* the: 5%
* to: 5%
* for: 30%
* be: 60%

The chance that I select the is 5%, to: 5%, for: 30%, and be: 60%. If we have a top\_p of 50%, we use the top two token probabilities in the distribution: for and be. This would change the for and be percentages to 33% and 67%, respectively.

**Note:** These API hyperparameters should not be adjusted simultaneously. If you would like to affect the determinism of your output, choose only one to adjust.



### **Token Repetition**

Another way to manage the output variation is by controlling the repetition of tokens.

The API affords us two hyperparameters that lower the probabilities of tokens already present in the output. These hyperparameters are presence\_penalty and frequency\_penalty. The effect from each is subtle, but they are used to control the output in very different ways.

##### **presence\_penalty**

The entire presence\_penalty will be applied to a token’s probability if that token has been selected at least once in the generated output.

Increasing the presence\_penalty can make the output branch into new topics related to the initial topic to avoid repeating tokens. This can be useful for research or brainstorming, where you may want the output to produce new topics related to the input.

##### **frequency\_penalty**

The strength of the frequency\_penalty corresponds to how often the token already appears in the output. A higher frequency\_penalty will reduce the likelihood of repeated words and phrases. This is useful if we look for an output with a variety of phrases while staying on topic.

The reduction in probability from these penalties depends on how likely the token was to be chosen initially. There are instances where tokens may still be the optimal choice despite the penalty.

LLMs provide varying outputs given the same input, so observing the effect of both hyperparameters can be a challenge. Getting proficient with these hyperparameters is a great way to guide your output to desired results.

**Instructions**

The GPT simulator gives you two options, each with the same text prompt. One of the prompts has no frequency or presence penalty applied, while the other has high penalties applied.  
With the given prompt about the Mona Lisa, you will notice that the two outputs cover much of the same content. The output with the higher penalties includes more varying content, such as the inclusion of King Francis I and more focus on attempted thefts.  
It is also noticeable that the higher penalty output uses more words to describe similar information to no penalty output. This could be the penalties leaning toward unused tokens to describe the same content.  
As mentioned, the effects of these penalties can be subtle, but with experience, you will learn how to apply them to receive the desired output with fewer iterations of inputs.  
There are several sample prompts to choose from, and you can use the arrows to select the next or previous prompt. Try submitting each prompt to see what the GPT simulator returns.  
Use the “Submit Prompt” button in the workspace to send your selected prompt!

### **Text Completion Response**

Once we create our input, tune the hyperparameters to guide the output, and send the request to the API, we must be ready to handle the data returned with the response. The response contains the following:

* **id**: Identifier unique to each completion
* **object**: Completion endpoint used
* **created**: Timestamp of the completion
* **model**: The model used for the completion
* **system\_fingerprint**: An identifier of the backend system configuration
* **choices**: List of completions, covered in depth below
* **usage**: Token usage, split up into input, output, and total tokens used

##### **choices**

choices contains a list of

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

completions. The default request will result in a single completion. choices contains the following:

* **index**: The listed completion number starting at 0.
* **message**: Contains an object with role with content parameters:
  + **role**: This will always be assistant in the response
  + **content**: The completion text
* **finish\_reason**: The reason why the completion finished. The options are:
  + **stop**: The completion is done
  + **length**: The completion reached the maximum length of tokens dictated by the request
  + **function call**: A function given in the response is called
  + **content filter**: The completion was flagged by the OpenAI content policy
  + **null**: The completion is not complete or in progress

This response message with the assigned "assistant" role can be combined with the "user" role in the input prompt. If a new "user" message is added, we can create a conversation. Here’s an example of a potential new input "messages":

"messages" = [

{

"role": "system",

"content": "Phrase all answers based on a 5th grade reading level."

},

{

"role": "user",

"content": "What is a perimeter?"

},

{

"role": "assistant",

"content": "The perimeter is the distance around the edge of a shape or an object."

},

{

"role": "user",

"content": "What is an area?"

}

Copy to Clipboard

The above messages combine the system and user messages from the original input, the assistant message from the response, and a new user message asking about area.

The question, “What is an area?” is a little vague and could generate responses that have nothing to do with the area of a shape. With the previous messages added to the input, the LLM will now have more context regarding how it might reply.

**Instructions**

The web browser contains the data representation of the chat/completions response.

### **Text Completion Response**

Once we create our input, tune the hyperparameters to guide the output, and send the request to the API, we must be ready to handle the data returned with the response. The response contains the following:

* **id**: Identifier unique to each completion
* **object**: Completion endpoint used
* **created**: Timestamp of the completion
* **model**: The model used for the completion
* **system\_fingerprint**: An identifier of the backend system configuration
* **choices**: List of completions, covered in depth below
* **usage**: Token usage, split up into input, output, and total tokens used

##### **choices**

choices contains a list of

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

completions. The default request will result in a single completion. choices contains the following:

* **index**: The listed completion number starting at 0.
* **message**: Contains an object with role with content parameters:
  + **role**: This will always be assistant in the response
  + **content**: The completion text
* **finish\_reason**: The reason why the completion finished. The options are:
  + **stop**: The completion is done
  + **length**: The completion reached the maximum length of tokens dictated by the request
  + **function call**: A function given in the response is called
  + **content filter**: The completion was flagged by the OpenAI content policy
  + **null**: The completion is not complete or in progress

This response message with the assigned "assistant" role can be combined with the "user" role in the input prompt. If a new "user" message is added, we can create a conversation. Here’s an example of a potential new input "messages":

"messages" = [

{

"role": "system",

"content": "Phrase all answers based on a 5th grade reading level."

},

{

"role": "user",

"content": "What is a perimeter?"

},

{

"role": "assistant",

"content": "The perimeter is the distance around the edge of a shape or an object."

},

{

"role": "user",

"content": "What is an area?"

}

Copy to Clipboard

The above messages combine the system and user messages from the original input, the assistant message from the response, and a new user message asking about area.

The question, “What is an area?” is a little vague and could generate responses that have nothing to do with the area of a shape. With the previous messages added to the input, the LLM will now have more context regarding how it might reply.

**Instructions**

The web browser contains the data representation of the chat/completions response.

| Response Object {  "id": "chatcmpl-001",  "object": "chat.completion",  "created": 1677652288,  "model": "gpt-3.5-turbo",  "system\_fingerprint": "fp\_0000000001",  "choices": [{  "index": 0,  "message": {  "role": "assistant",  "content": "The perimeter is the distance around the edge of a shape or an object.",  },  "logprobs": null,  "finish\_reason": "stop"  }],  "usage": {  "prompt\_tokens": 19,  "completion\_tokens": 15,  "total\_tokens": 34  }  } |
| --- |

### **Summary**

Congratulations on completing the OpenAI API lesson! LLMs are powerful AI tools, and to harness their power well and create efficient AI applications, it is important to understand how the APIs are used to generate

[text](https://www.codecademy.com/resources/docs/ai/generative-ai/text)

Preview: Docs Loading link description

outputs desired by the user. Knowing the API hyperparameters’ role, functionality, and limitations ters is also immensely important in

[prompt engineering](https://www.codecademy.com/resources/docs/ai/prompt-engineering)

Preview: Docs Prompt Engineering is the study and implementation of prompts which are used to instruct an Artificial Intelligence (AI) into completing a specific task.

.

In this course, we covered the following:

* The API provides many endpoints with different functionality, including text completion, image generation, and audio transcription.
* OpenAI has many different models of varying performance and cost attached to different endpoints.
* Tokens are the pieces of words the models operate on, and model usage cost is based on the number of tokens used.
* Crafting an input prompt is a trial and error process known as prompt engineering.
* temperature and top\_p are two parameters that control the token probabilities and affect the determinism of the model output.
* presence\_penalty and frequency\_penalty are two parameters that affect how varied the output content is, from subtle decreases in repetition to major branches in the output topic.
* The API response contains the completion text and other information such as ID, completion timestamp, finish reason, and token usage.

# Next Steps

**What’s next?**

Congratulations on completing Intro to OpenAI GPT API! In this course, you learned about large language models and bringing AI into your code, and are now able to:

* Understand what APIs are and their uses
* Create reliable output from LLMs
* Write effective input prompts

Now that you have finished your learning journey, you may be asking, “What’s next?” Here are our recommendations for the next steps:

### [**Intro to Language Models in Python**](https://www.codecademy.com/learn/intro-to-language-models-in-python)

Want to learn more about large language models, the technology behind GPT? The Intro to Language Models in Python course is a great way to gain solid foundational knowledge of large language models.

### [**Intro to ChatGPT**](https://www.codecademy.com/enrolled/courses/intro-to-chatgpt)

ChatGPT is a large language model created by OpenAI with a wide range of potential applications for business, engineers, and daily life.

### [**Unit Testing with Generative AI Case Study**](https://www.codecademy.com/learn/unit-testing-with-generative-ai-case-study)

Practice using AI to create a Python unit test.

### [**Pair Programming with ChatGPT Case Study**](https://www.codecademy.com/learn/pair-programming-with-chat-gpt-case-study)

Practice using ChatGPT as a pair-programming buddy.

Once again, congratulations on finishing your course! We’re excited to see what you accomplish next.

**23 Using OpenAI API**

Open**AI** for Python : <https://platform.openai.com/docs/overview>

**Create and export an API key**

<https://platform.openai.com/docs/quickstart>

* Virtual environment
* Api keys (windows, iOS, agnostic)
* Pip install

# Using OpenAI APIs: Accessing OpenAI APIs from Python

OpenAI application programming interfaces (APIs) represent a groundbreaking leap in the accessibility of state-of-the-art natural language processing (NLP) capabilities. These APIs provide developers with a powerful toolset to integrate advanced language models seamlessly into their applications, products, and services. You will start this course by engaging with OpenAI through the command-line, utilizing the OpenAI APIs. You will learn how to authenticate yourself using API keys when programmatically accessing API endpoints using cURL commands. You will explore how to configure context for past interactions with the model and access both chat completions and legacy completions APIs via their respective endpoints. Moving onto Python, you will install the OpenAI library to create a client object for endpoint access. You will configure the API key and send requests to the chat completions endpoint with prompts in the JSON format. You will also explore the legacy completions API using the same client object. You will be introduced to the diverse range of model offerings from OpenAI and learn how to use those models. Finally, you will configure model parameters to adjust the response from the model. You will learn about the seed parameter to receive deterministic responses and how the system fingerprint helps track infrastructure changes on the server. You will explore various parameters, including Top P and Temperature for controlling creativity, max length, and stop sequences for response length, and frequency and presence penalty for word and topic repetition.

## Table of Contents

[1. Video: Course Overview (it\_aiopapdj\_02\_enus\_01)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_0)

[2. Video: Generating Application Programming Interface (API) Keys to Use with Endpoints (it\_aiopapdj\_02\_enus\_02)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_1)

[3. Video: Making API Requests with cURL (it\_aiopapdj\_02\_enus\_03)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_2)

[4. Video: Making Our First API Request Using Python (it\_aiopapdj\_02\_enus\_04)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_3)

[5. Video: Exploring Chat Completions and JSON Mode (it\_aiopapdj\_02\_enus\_05)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_4)

[6. Video: Streaming Chat Completions with the Legacy Completions API (it\_aiopapdj\_02\_enus\_06)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_5)

[7. Video: Exploring Chat and Completion Models (it\_aiopapdj\_02\_enus\_07)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_6)

[8. Video: Setting up Utility Functions to Explore Model Determinism (it\_aiopapdj\_02\_enus\_08)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_7)

[9. Video: Using the Seed Parameter for Almost Deterministic Responses (it\_aiopapdj\_02\_enus\_09)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_8)

[10. Video: Tweaking Model Parameters – Temperature and Max Tokens (it\_aiopapdj\_02\_enus\_10)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_9)

[11. Video: Tweaking Model Parameters – Stop Sequences Top P and Penalties (it\_aiopapdj\_02\_enus\_11)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_10)

[12. Video: Course Summary (it\_aiopapdj\_02\_enus\_12)](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_11)

[Course File-based Resources](https://cdn2.percipio.com/secure/c/1763531837.1a5666a9461288cd38a95fba92d693be7e872fdc/eot/transcripts/59fbfcf2-3772-4bde-9b31-e9ed667d8159/it_aiopapdj_02_enus.html#section_12)

## 1. Video: Course Overview (it\_aiopapdj\_02\_enus\_01)

****

In this video, we will discover the key concepts covered in this course.

* *discover the key concepts covered in this course*

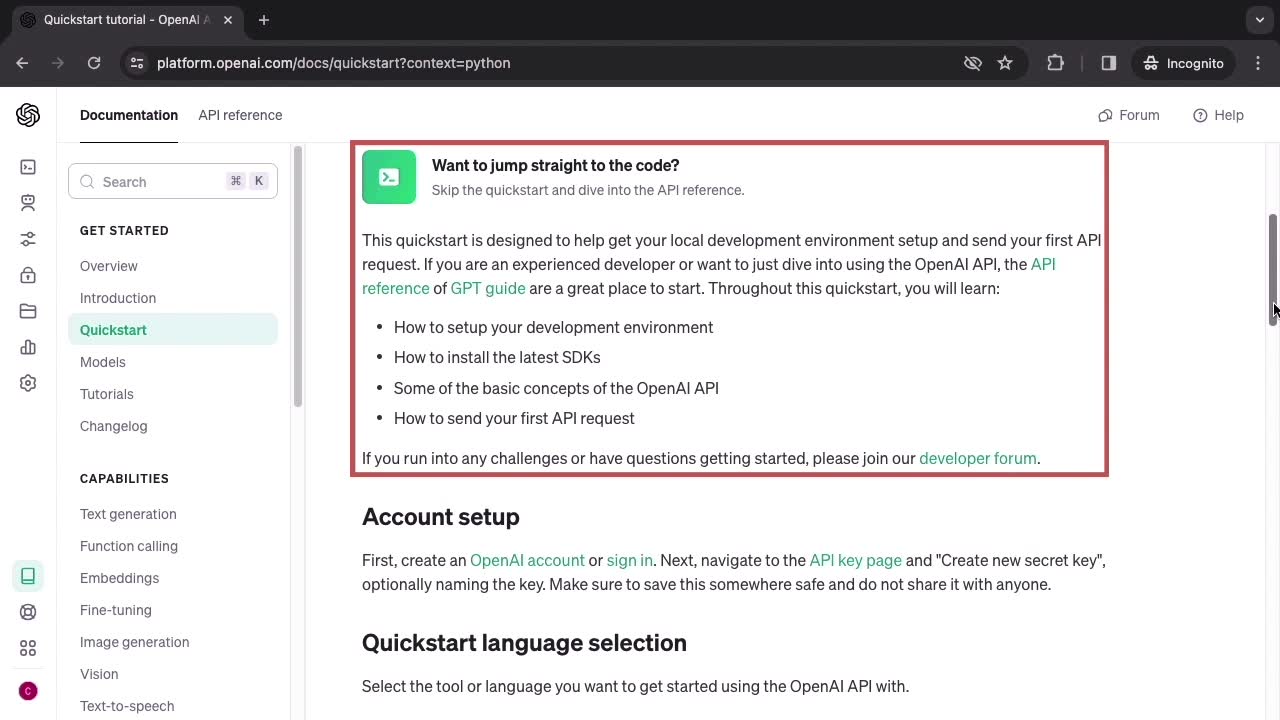
[Video description begins] *Topic title: Course Overview. Presented by: Janani Ravi.* [Video description ends]

Hi and welcome to this course, accessing OpenAI APIs from Python. My name is Janani Ravi, and I'll be your instructor for this course. OpenAI APIs allow you to access the state-of-the-art natural language processing capabilities that OpenAI has to offer. These APIs provide developers with a powerful toolset to integrate advanced language models seamlessly into their applications, products, and services. You'll start this course by acquiring the skills to engage with OpenAI through the command-line, utilizing the OpenAI APIs.

You'll learn how to authenticate yourself using API keys when programmatically accessing API endpoints using cURL commands. You'll explore how you can configure context for past interactions with the model. You'll access both the chat completions and legacy completions APIs via their respective endpoints. Moving on to Python code, you'll install the OpenAI Python libraries, which you'll then use to instantiate a client object for accessing endpoints. You'll configure the API key and send requests to the chat completions endpoint, using prompts in the JSON format. You'll also explore the legacy completions API using the same client object. You'll be introduced to the diverse range of model offerings from OpenAI and learn how to use these models.

Finally, you turn your attention to configuring model parameters to tweak the response from the model. You'll learn about how you can use the seed parameter to get deterministic responses from the model and how you can use the system fingerprint to track infrastructure changes on the model server. You'll explore various parameters, including Top P and Temperature for controlling models' creativity, max length and stop sequences for controlling the response length, and frequency and presence penalty for word and topic repetition. In conclusion, this course will equip you with a solid understanding of accessing and manipulating OpenAI APIs using Python.

## 2. Video: Generating Application Programming Interface (API) Keys to Use with Endpoints (it\_aiopapdj\_02\_enus\_02)

****

Learn how to create application programming interface (API) keys for OpenAI APIs.

* *create application programming interface (API) keys for OpenAI APIs*

[Video description begins] *Topic title: Generating Application Programming Interface (API) Keys to Use with Endpoints. Presented by: Janani Ravi.* [Video description ends]

At this point, we've explored the APIs that OpenAI has to offer using the OpenAI Playground, which means we haven't directly interacted with the APIs. Instead, we've used a nice graphical user interface that allowed us to play around with different models and tweak model parameters. From this demo onwards, we'll work directly with the OpenAI APIs using the REST endpoints to interact with the various models that OpenAI has to offer. Now there's a little bit of setup that you have to do in order to be able to hit those REST API endpoints, and exactly what setup you need to do, is available here in this quickstart Link, platform.openai.com/docs/quickstart. This is where you'll see how you can download the SDK and setup API access from your local machine.  
  
[Video description begins] *A page called Quickstart tutorial displays. It contains 2 sections: Documentation and API reference. The Documentation sections contains 2 headers: GET STARTED and CAPABILITIES. The GET STARTED header consists of the following options: Overview, Introduction, Quickstart, Models, Tutorials, and Changelog. The CAPABILITIES header contains the following options: Text generation, Function calling, Embeddings, Fine-tuning, Image generation, and so on. The Quickstart option is now highlighted. The main pane displays the procedure of running the OpenAI API, Account setup, Quickstart language selection underneath Developer quickstart.* [Video description ends]

Here you can see what the quickstart will help us achieve. Setting up our development environment, installing the latest SDKs, we'll cover some basic concepts of the OpenAI API, and how to send the first API request. Well, that's what we'll be doing in this demo. I just want to show you where the document is, so that you're familiar with the documentation, and if you ever need to reference it for anything extra that you want to do beyond the demos that we'll cover, well, you know where to find the docs. We've already covered the first step here in this process. We've created an OpenAI account, and we've already signed into that account.

Next, based on the technology that we're going to use, to hit the APIs and get the responses, you'll need to setup for that technology. Observe, there are three options here, curl, Python, and Node.js. In this demo, we'll first see how we can access the APIs using the curl command line utility, and then we'll work with Python. We won't be working with Node.js.  
  
[Video description begins] *The Quickstart language selection header is highlighted. It contains the following options: curl, Python, and Node.js. The Python option is selected. It contains 3 steps: Setup Python, Setup your API key, and Sending your first API request. The Setup python step contains the following headers: Install Python, Setup a virtual environment (optional), and Install the OpenAI Python library. The Setup your API key step contains the following headers: Setup your API key for all projects (recommended) and Setup your API key for a single project.* [Video description ends]  
  
I currently have the Python option selected and you can see that OpenAI provides a custom Python library which makes working with the OpenAI API very simple and efficient.

Now in order to work with Python, the first step you have to do is Install Python, but hopefully, you already have Python installed on your local machine. Notice here at the bottom, OpenAI specifies that you need at least Python 3.7.1 or newer. So, make sure that you have a recent version of Python. Now, in order to work with OpenAI in Python, it's a best practice to setup a virtual environment. Setting up the virtual environment is completely optional, but it's a best practice because then all of the packages and libraries that you install in your virtual environment will be isolated from your system-wide installation.

When we work with Python, we'll definitely setup the virtual environment and I'll show you the steps involved in that. Within the virtual environment, we'll Install the OpenAI Python library using a simple pip install. Now, in order to be able to access the OpenAI REST endpoints and so that all of the billing occurs to your account, you need to setup an API key. This API key is generated on the OpenAI platform using your OpenAI account and is sent with every request that you make to the OpenAI APIs. You can configure this API key as an environment variable in your Windows or Mac machine.

Once you set up the environment variable, every request that you make from the machine where this is configured, well, that request will be authenticated to the OpenAI endpoints. Here on this page, you'll find instructions for how you can setup your environment variable whether you're on a MacOS device or a Windows device, and you're actually free to follow the instructions here.   
  
[Video description begins] *The Step 2 Setup your API key is now highlighted. The Python library displays 2 files: MacOS and Windows. The MacOS file requires to follow the following steps: Open Terminal, Edit Bash Profile, Add Environment Variable, Save and Exit, Load Your Profile, and Verification. The Windows file requires to follow the following steps: Open Command Prompt, Set environment variable in the current session, Permanent setup, and Verification.* [Video description ends]  
  
I'll show you how you can generate an API key in just a bit, and after that, you can follow these steps that you see here on screen, If you're on a MacOS device, to setup your API key as a part of your environment. Observe that the name of the environment variable that you have to configure is OPENAI\_API\_KEY.

Now, let's say you're not on a MacOS machine, instead you're working on a Windows machine. You can follow the steps here under the Windows section to set up your API key as an environment variable. I'll be doing something a little different in our curl and Python demos. I'll use the OS package that Python offers to setup the API key as my environment variable. I like following that process because it's device-agnostic. So, whether you're on MacOS or Windows, setting up the API key programmatically in Python will work on both operating systems. This API key is needed whether you're working in Python or in curl.

Now, if you want instructions for how you can use the API endpoint with curl and what you need to setup, select curl as an option here on this page. It's quite likely that you already have curl installed on your machine, but if you don't, here are instructions to get setup with curl whether you're on Windows or MacOS. Now that I've given you the big picture of getting installed and setup with OpenAI, you can simply follow along with me, and we'll first see how we can work with the OpenAPI REST points using curl.

Now, if you don't have curl, head over to this URL, everything.curl.dev/get. Here you'll find instructions for installing curl on Linux, Windows, macOS, and even within a Docker environment.  
  
[Video description begins] *The page called Install curl appears underneath Everything curl. The left pane contains the following options: Introduction, How to read this book, The cURL project, Network and protocols, Install curl, Source code, Build curl, Command line basics, and so on. The main pane displays various sources of curl: Linux, Windows, macOS, and Docker.* [Video description ends]  
  
If you're on a macOS machine, it's quite likely that you already have curl pre-installed, and maybe that's true for later versions of Windows as well. Now, since curl is a fairly simple and straightforward utility and something that you're likely familiar with, I won't go through the curl installation process. Instead, let's head over to the terminal and see how we can use the OpenAI APIs using curl.

Here I am in the terminal window of my Mac machine.  
  
[Video description begins] *The terminal window appears. It displays the following message: curl --version.* [Video description ends]  
  
I already have curl installed and available. When I run curl --version, you can see that I'm using curl 7.87.0. That's the recent version. Before we make a single request, we need an API key to authenticate ourselves to OpenAI.  
  
[Video description begins] *The page called Overview appears underneath OpenAI API. The Documentation section is highlighted again. The Overview option is selected from the left. The main pane contains the following headers underneath Welcome to the OpenAI developer platform: Start with the basics and Watch the first OpenAI Developer Day keynote.* [Video description ends]  
  
Head over to the OpenAI Platform and open up the left sidebar and here you'll find an option for API keys.  
  
[Video description begins] *The corresponding page called API keys - OpenAI API appears. The left pane contains the following options: Playground, Assistants, Fine-tuning, API keys, Files, Usage, Settings, Documentation, Help, All products, and so on. The API keys option is now selected. The main pane displays the following columns: NAME, KEY, CREATED, and LAST USED. Below, it contains a header called Default organization.* [Video description ends]  
  
Select this option and this will take you to a page where all of the API keys that you've previously created should be listed. You can see here that API keys are a secret and should not be shared with anyone. The API key that I'm about to create, I'm going to display it to you, but I'm going to get rid of it. Delete it as soon as I'm done with all of the demos. You can see that I've previously created an API key that I have used that's listed here.

I'm going to create a new secret key to use for our demos. Click on this button Create new secret key.  
  
[Video description begins] *A pop-up box appears with the heading Create new secret key. It contains a field called Name.* [Video description ends]  
  
The only thing you need to specify is the Name of this API key, that's optional, but I suggest you give it a meaningful name so, you know why exactly this key is being used. I've called this key learning-openai-apis and I click on Create secret key and this will display the API key to you for the first and last time. This is the time when you need to copy this key and store it safely somewhere because you won't be able to view this key again. So, make sure you copy this key and store it somewhere so that we can now use it to make API requests. Once you click on Done, you'll see that this newly created secret key is available here in this API keys list.

## 3. Video: Making API Requests with cURL (it\_aiopapdj\_02\_enus\_03)

****

Find out how to send an API request from cURL.

* *send an API request from cURL*

[Video description begins] *Topic title: Making API Requests with cURL. Presented by: Janani Ravi.* [Video description ends]

Now the key that I copied over is the one that I'm going to use to authenticate myself when I make API request to the OpenAI endpoints.  
  
[Video description begins] *A page called API keys - OpenAI API appears. The left pane contains the following options: Playground, Assistants, Fine-tuning, API keys, Files, Usage, Settings, Documentation, Help, All products, and so on. The API keys option is now selected. The main pane displays the following columns: NAME, KEY, CREATED, and LAST USED. Below, it contains a header called Default organization.* [Video description ends]  
  
Now the very first request I'm going to make is to hit the endpoint that allows us to access all of the models available in OpenAI. Notice I use the curl command. By default, this will be a GET request. The endpoint is at https://api.openai.com/v1/models.

Notice how the version of the API is part of the REST endpoint. As new versions of the endpoint are released, we should move on to v2, v3, and so on. -H allows me to specify a header with my request and I have a Authorization header with a Bearer token and this token is the API key that we just generated and I copied over.  
  
[Video description begins] *A new terminal pane appears. The following command is added. Line 1 reads: curl https://api.openai.com/v1/models \. Line 2 reads: -H "Authorization: Bearer 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".* [Video description ends]  
  
Now, without having this Authorization header, you won't be able to access this endpoint. Let's hit Enter and you can see that we get a JSON response.

This JSON response returns all of the models available with OpenAI. For every model we have a unique model id. For example, gpt-4-0613 is a recent GPT-4 model. The object property is used to indicate what kind of object was returned from the server. You can see all of the objects here are models. Objects can also be, say, chat completions or an image that was generated. The created attribute gives you the timestamp when the model was created, and you can see that the model is owned by openai. And here you'll see all of the gpt-4, gpt-3, and older models that OpenAI currently supports. You can see that most of the models are owned by either openai or openai-dev.

Now here at the very end, you see that there are some models here that may not be in your list. These are the models that start with the prefix ft: these are the fine-tuned models that I was playing around with. These are not models that will be in the list that you retrieve. If you want to retrieve information about just a single model, you simply specify the ID of the model as a part of your URL endpoint. Notice my endpoint api.openai.com/v1/models and then the ID of the model whose details I want to retrieve gpt-4-0613 and I have the same Authorization Bearer token in the header.   
  
[Video description begins] *The following command is added. Line 1 reads: curl https://api.openai.com/v1/models/ gpt-4-0613 \. Line 2 reads: -H "Authorization: Bearer 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".* [Video description ends]  
  
Once you hit Enter, you'll retrieve the details of exactly one model, and that's what you get here in the response.

Let's see how we can specify a prompt using the API endpoint and get a response to our query. If you remember from the OpenAI Playground, there are two different chat modes. There is the legacy completions mode and there is the current chat mode. Now, in order to use the chat mode, the API endpoint ends with the path/chat/completions, and you can see that this is where I make my curl request api.openai.com/v1/chat/completions.  
  
[Video description begins] *The following lines are now added. Line 1: curl https://api.openai.com/v1/chat/completions \. Line 2 reads:-H "Content-Type: application/json" \. Line 3 reads:-H "Authorization: Bearer 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" \. Line 4 reads:-d '{. Line 5 reads:"model": "gpt-3.5-turbo",. Line 6 reads:"messages": [. Line 7 reads:{. Line 8 reads: "role": "user",. Line 9 reads: "content": " Could you recommend some resources I could use to learn Python programming?". Line 10 reads:}. Line 11 reads:]. Line 12 reads:}' .* [Video description ends]  
  
That is the API URL endpoint.

I have some headers here, the first one is Content-Type: application/json. That's because I'm specifying the request body in the JSON format. I have the second header that's required, the one with the Authorization token. I use the -d flag in curl to specify the request body and then the JSON that follows is basically the structure of the JSON request that the OpenAI endpoint expects. Just a heads up that the minute you specify a request body as a part of your request, curl will make a POST request to this HTTP endpoint. You need to specify the model that you want to use that will generate the response to your request and in the messages field you'll specify your message.

Observe the square bracket after messages, you can specify a list of messages so you can specify back and forth conversation between the system and the user. I just have a single message here in this list. Notice that every message is a JSON entity and for each message, you'll specify the role of the person sending the message. Is it the system that is the assistant or the user? Here I send the message to the API endpoint in the role of a user and the content field contains the actual contents of my prompt. "Could you recommend some resources I could use to learn Python programming?"

Please just take a moment to observe the structure of this JSON request. This is the structure we'll be using whenever we work with the natural language models. Go ahead and hit Enter, and here is the response from the server. Notice that every response has a unique id and because we're using the chat completions API, the id starts with chatcmpl. The object property tells you what kind of object was returned, you can see it's of type chat completion. You can see when it was created, the model that was used.

Notice the model specifies exactly which model was used. We asked for gpt-3.5-turbo, that's in the model property in our request, but the actual model that was used to generate this response was gpt-3.5-turbo version 0613. And then you have the choices field, which is just an array of messages returned as a part of the response. The index gives you the index position of the message, the message property, the actual response. Notice that every message has a role.

The role says assistant because it's the assistant responding to our query. The actual content contains the actual response. Resources that can be used to learn Python programming. Let's look at some of the other properties here in the response. Notice the finish\_reason is stop. This indicates why the model stopped generating the response because it felt that the response was complete. Sometimes the response can end abruptly if you've run out of the max length tokens that you've specified for the response. You also get usage metrics for this particular request and response from the API. The prompt took up 20 tokens.

The completion that is the response from the model took up 365 tokens, leading to a total token usage of 385. Don't worry about the other fields here, logprobs and system\_fingerprint, we'll understand those in due course. Let's make another API request. Once again, this is to the chat completions API. You can see v1/chat/completions on the very first line. The headers are exactly as before, the Content-Type and Authorization headers. I continue to use the gpt-3.5-turbo model.

Observe that I've now specified two messages in the messages list, and the messages have two different roles.  
  
[Video description begins] *The following lines are now added. Line 1: curl https://api.openai.com/v1/chat/completions \. Line 2 reads:-H "Content-Type: application/json" \. Line 3 reads:-H "Authorization: Bearer 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" \. Line 4 reads:-d '{. Line 5 reads:"model": "gpt-3.5-turbo",. Line 6 reads:"messages": [. Line 7 reads:{. Line 8 reads: "role": "system",. Line 9 reads: "content": " You are a teaching assistant for a Math course". Line 10 reads:},. Line 11 reads:{. Line 12 reads: "role": "user",. Line 13 reads: "content": " How do I solve 3x + 5 = 32? Could you give me step by step instructions?". Line 14 reads:}. Line 15 reads:]. Line 16 reads:}'.* [Video description ends]  
  
The first message is for the system role, that is the role that we assign to the model, that is a part of the system input that we specified in the OpenAI Playground. That message says, "You are a teaching assistant for a Math course". So, this is the role or persona that I've applied to the model or the system. The second message has role user, and this is where I ask my question. "How do I solve 3x + 5 is equal to 32? Could you give me step-by-step instructions?" Let's go ahead and submit these messages to the API endpoint and here is the response from the model and you can see that it gives me step-by-step instructions.

Rest of the structure of the response is exactly what we've seen before. Notice the model property gives us the exact version of the model used and the response message comes back with the role of assistant. And you can see the usage metrics here at the bottom, total\_tokens 196. Let's look at another example, again with the chat completions endpoint, observe that I've now switched over to using a newer model, gpt-4. The messages follow the same structure as before.  
  
[Video description begins] *The following lines are now added. Line 1: curl https://api.openai.com/v1/chat/completions \. Line 2 reads:-H "Content-Type: application/json" \. Line 3 reads:-H "Authorization: Bearer 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" \. Line 4 reads:-d '{. Line 5 reads:"model": "gpt-4",. Line 6 reads:"messages": [. Line 7 reads:{. Line 8 reads: "role": "system",. Line 9 reads: "content": " You are a sarcastic teenager who is bored with everything in life talking to an adult". Line 10 reads:},. Line 11 reads:{. Line 12 reads: "role": "user",. Line 13 reads: "content": " How are things with you? Have you figured out what colleges you want to apply to?". Line 14 reads:}. Line 15 reads:]. Line 16 reads:}'.* [Video description ends]  
  
Every message has a role and content.

The system role allows me to give instructions to the model. "You are a sarcastic teenager who is bored with everything in life, talking to an adult". This is the persona of the model or my instructions to the model. The message with the role user. That's my prompt. "How about things with you? Have you figured out what colleges you want to apply to?" Let's see what our sarcastic teenager has to say. You can see the response message here. The content contains the actual response. Oh wow, what an original question that I've definitely never heard of before, and so on. It's definitely a very sarcastic response.

Now let me show you one last example with the chat completions API and here I'll discuss why exactly messages is an array. Having messages be an array allows you to give instructions to the model that you do in the role of system, that's the first message that you can see here on screen.   
  
[Video description begins] *The following lines are now added. Line 1: curl https://api.openai.com/v1/chat/completions \. Line 2 reads:-H "Content-Type: application/json" \. Line 3 reads:-H "Authorization: Bearer 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" \. Line 4 reads:-d '{. Line 5 reads:"model": "gpt-4",. Line 6 reads:"messages": [. Line 7 reads:{. Line 8 reads: "role": "system",. Line 9 reads: "content": " You are a helpful assistant.". Line 10 reads:},. Line 11 reads:{. Line 12 reads: "role": "user",. Line 13 reads: "content": "Who won the cricket world cup in 2011?". Line 14 reads:},. Line 15 reads:{. Line 16 reads: "role": "assistant",. Line 17 reads: "content": "India won the cricket world cup in 2011.". Line 18 reads:},. Line 19 reads:{. Line 20 reads:"role": "user",. Line 21 reads: "content": "Where was it played? Do you know the details of the final match?". Line 22 reads:}. Line 23 reads:]. Line 24 reads:}'.* [Video description ends]  
  
The system message here on screen is simply "You are a helpful assistant." In addition to system messages, you can also simulate context history or back-and-forth conversation that you've previously had with the model. When the model has access to your conversation history, it's able to follow up with responses that take into account this context.

Here, I've simulated a back-and-forth with the model. My first prompt in the role of the user says, "Who won the cricket world cup in 2011?" Let's say I've asked this question and the assistant responded, "India won the cricket world cup in 2011." Now when I ask a follow-up question, I don't need to provide the entire context. In the role of the user, I ask a follow-up question. "Where was it played? Do you know the details of the final match?" Notice that this prompt does not talk about the cricket world cup, does not talk about 2011, does not talk about India winning. The entire context is present as a part of the previous messages in the messages array.

Now the response to the last question will still be meaningful. So, here is what the response looks like. It talks about the 2011 World Cup held in India, Sri Lanka, and Bangladesh. The final match was on 2nd April 2011 at Wankhede Stadium and a whole bunch of other details. Now let's say you wanted to further follow up on this conversation. You take this response from the model, append it to the list of messages that you had previously, and send the entire list to the model. This is how you simulate conversation using the API.

## 4. Video: Making Our First API Request Using Python (it\_aiopapdj\_02\_enus\_04)

****

Discover how to send an API request from Python.

* *send an API request from Python*

[Video description begins] *Topic title: Making Our First API Request Using Python. Presented by: Janani Ravi.* [Video description ends]

At this point, you have a small taste of working with the OpenAI APIs, using curl to make requests, and viewing responses in the console. For the rest of this course, we'll work with the OpenAI APIs using their Python library. So, let's get set up with the virtual environment within which we'll install the OpenAI Python package and we'll run Jupyter Notebooks to run code in Python. I'm going to create a new directory called projects, that will serve as my current working directory.  
  
[Video description begins] *A terminal window appears. It displays the following commands. Line 1 reads: mkdir projects. Line 2 reads: cd projects/.* [Video description ends]  
  
I've created it using the mkdir command and I'm going to cd into projects.

Let's check the version of Python that I have installed on my local machine. I run python --version and you can see that I'm working with Python version 3.10.9.   
  
[Video description begins] *The command now reads: python --version.* [Video description ends]  
  
Remember that your Python version needs to be at least 3.7.1 before you can work with OpenAI APIs using their Python package. We're going to be running our Python code on Jupyter Notebooks. Let me run jupyter --version and show you that I have a recent version of the Notebook Server installed.  
  
[Video description begins] *The command now reads: jupyter --version.* [Video description ends]  
  
Any recent version of the Jupyter Notebook server should suffice.

Here in my projects folder, I'm going to create another subfolder called open\_ai\_apis and cd into the subfolder. This is where I'll place all of the notebooks with my code. I'm going to run all my Python code within a Python virtual environment.  
  
[Video description begins] *The command 1 reads:mkdir open\_ai\_apis. The command 2 reads: cd open\_ai\_apis.* [Video description ends]  
  
A Python virtual environment is a self-contained directory or folder that contains a separate Python interpreter and its own set of libraries and packages. Allows you to create isolated environments for different Python projects, each with its own dependencies, without interfering with the system by Python installation.  
  
[Video description begins] *The command reads: python3 -m venv openai\_venv.* [Video description ends]  
  
I'm going to use the python3 -m venv command to create a virtual environment named openai\_venv. That's the virtual environment within which we'll run our code.

Go ahead and run this command and this will create a new directory with the same name as that of the virtual environment, openai\_venv, ls-l will show you that this directory has now been created.  
  
[Video description begins] *The command now reads: ls -l.* [Video description ends]  
  
In order to use the virtual environment, we'll first have to activate it, and on macOS, this is the command that you run to activate the virtual environment, source openai\_venv, or whatever it is that you've called your virtual environment /bin/activate.   
  
[Video description begins] *The command now reads: source openai\_venv/bin/activate.* [Video description ends]  
  
If you're running on a Windows machine, the command that you use to activate the virtual environment is the one that you see here on screen, openai\_venv\Scripts\activate.bat.

This will run the batch script that you need to activate the virtual environment and once you execute this, you'll find that the prompt is updated to indicate that you're now within the openai\_venv. Now in order to be able to work with Jupyter Notebooks within this virtual environment, we need to install a specific package and that is ipykernel. This is a Python package that provides the communication between the Jupyter Notebook interface and the Python kernel.  
  
[Video description begins] *The command reads: pip install ipykernel.* [Video description ends]  
  
The Python kernel is just the engine that executes our Python code. We want the kernel to be able to use the Python that's available within our virtual environment and that's why we need to install ipykernel.

Once ipykernel has been installed, we can check to see what kernels are available to our Jupyter Notebook server. Run jupyter kernelspec list and this will list out all of the kernels that Jupyter Notebook has available to use.  
  
[Video description begins] *The command highlights: jupyter kernelspec list.* [Video description ends]  
  
And you can see there is just one kernel here, the python3 kernel. This is the system-wide installation of Python and not the Python that runs within our virtual environment. In order to make the Python within our virtual environment available as a kernel for our Jupyter Notebook server, you need to run this command, python -m ipykernel install, let's install only for the single user and the name of the kernel is openai\_venv, that is our virtual environment.  
  
[Video description begins] *The next command reads: python -m ipykernel install --user --name=openai\_venv.* [Video description ends]

Once you've installed the Python in our virtual environment as a kernel, if you run jupyter kernelspec list once again, you'll find that our virtual environment is now available to our notebook server. Notice the available kernel says openai\_venv. We'll be using the pip package manager to install packages, so, let's upgrade our pip to the latest version available. The pip running within the virtual environment is an older version, so, I'll upgrade to use the latest version 22.3.1.  
  
[Video description begins] *The command reads: pip install --upgrade pip.* [Video description ends]

Let's go ahead and install the openai Python library and you can get this with a simple pip install, pip install openai will install the latest version of the OpenAI library on your local machine.   
  
[Video description begins] *The following command is highlighted: pip install openai.* [Video description ends]  
  
Now, if you have older versions from before November 2023, make sure that you upgrade your OpenAI library. The APIs underwent a significant change around that time, so you have to ensure that you're using the latest version.  
  
[Video description begins] *The following command is highlighted: openai --version.* [Video description ends]  
  
If you run openai --version, you'll see that we are using the latest version at the time of this recording, openai 1.3.9.

Now there are some other Python packages that we'll be using to visualize, maybe work with data just for small utility functions that we might write.  
  
[Video description begins] *The command reads: pip install pandas matplotlib.* [Video description ends]  
  
So, I'm going to pip install some of those packages, pandas, and matplotlib to start off with. As and when we need other packages, I'm going to go ahead and install them within the notebook itself. Now with this done, we are ready to start our notebook server.  
  
[Video description begins] *The new command is added: jupyter notebook.* [Video description ends]  
  
Run jupyter notebook and this should bring up the Notebook Server on port 8888. The Notebook Server homepage should automatically open up within a browser window. If it does not, you can simply copy and paste this URL in your browser. From here on in, we'll write all our code within Jupyter Notebooks running in our virtual environment.

Let me show you how the very first notebook is set up. For the remaining notebooks, we'll start off with the notebook open.  
  
[Video description begins] *A new Home Page appears in Jupyter. It contains the following sections: Files, Running, and Clusters. The Files section contains a list of items and the following tabs: Upload and New. The New tab displays the following Notebooks: Python 3(ipykernel), openai\_venv, Text File, Folder, and Terminal.* [Video description ends]  
  
If you click on New, you should find that you have two kernels available, the Python 3 kernel and the openai\_venv kernel. You can select the openai\_venv kernel right from within here in the new dialog or if you get started with the Python 3 kernel as I'm about to do here, I'll show you how you can change the kernel that you're running with from within the notebook. Before I change the kernel, let me just give this notebook a meaningful name.

Now in the very first demo using notebooks, I'm going to be running BasicPromptsUsingAPIs, go ahead and rename the notebook to have this name. I then use the View option to turn on Line Numbers.  
  
[Video description begins] *The Jupyter file named as BasicPromptUsingAPIs displays. It further displays a toolbar with the following tabs: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The View tab is selected and it contains the following options: Toggle Header, Toggle Toolbar, Toggle Line Numbers, and Cell Toolbar.* [Video description ends]  
  
Select the Toggle Line Numbers option so that line numbers are visible in each code cell. I'm also going to Toggle the Header and Toggle the Toolbar as well so that we have the entire screen for our code. On the top right of the screen, you can see that we are currently using the Python 3 kernel, this is the system-wide Python installation.

In order to run with the kernel in the virtual environment, head over to kernel, and in this drop-down, choose the Change kernel option and pick the openai\_venv kernel. With this change, your notebook is now running with the openai\_venv virtual environment.   
  
[Video description begins] *The Kernel tab is now active. It displays the following options: Interrupt, Restart, Restart & Clear Output, Restart & Run All, Reconnect, Shutdown, and Change kernel. The Change kernel contains the following sub-options: Python 3(ipykernel) and openai\_venv.* [Video description ends]  
  
Let's go ahead and get started coding. I'll show you how you can set up the API key and make your very first request using Python. Here are the two imports that we'll set up for every notebook that we'll run in this learning path, from openai import the openai object that is OpenAI with O capital and A and I both capital, also import the built-in os library in Python.  
  
[Video description begins] *The Input Cell is now added in the main pane. Line 1 reads: from openai import OpenAI. Line 2 reads: import os.* [Video description ends]  
  
Now go ahead and hit Shift+Enter to execute this and now let's set up the environment variable using os.environ.

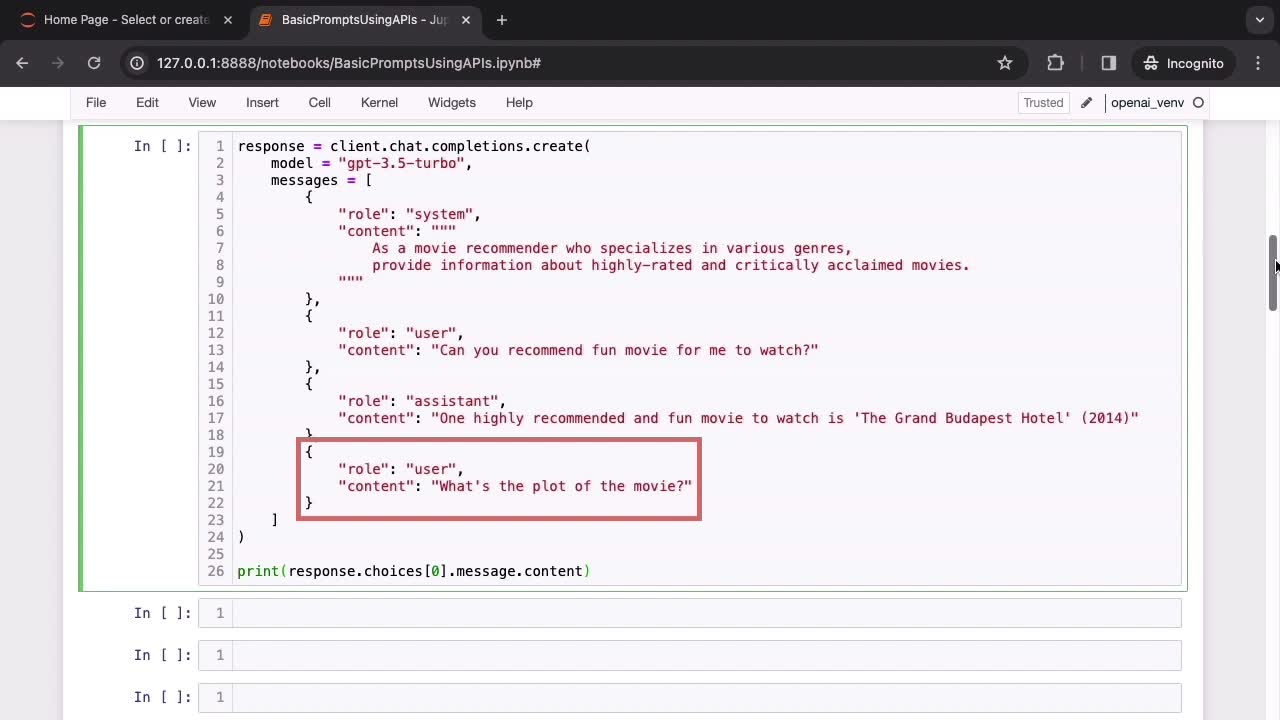
In order to be able to make requests from your local machine to the OpenAI API endpoints, you need to have the API key configured as an environment variable. Instead of setting it up as a system-wide environment variable, I'm setting it up for this particular project using os.environ. OPENAI\_API\_KEY is the name of the environment variable, and the value should be equal to the key that you generated using your account.  
  
[Video description begins] *The Input Cell 2 reads: os.environ["OPENAI\_API\_KEY"] = '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'.* [Video description ends]  
  
Once you've configured the key as a part of your environment, you should be able to instantiate the OpenAI client using client is equal to OpenAI.

Now this will not be possible if your environment doesn't have the OPENAI\_API\_KEY.  
  
[Video description begins] *The Input Cell 3 is now highlighted. Line 1 reads: client = OpenAI().* [Video description ends]  
  
Having instantiated the client, we are now ready to make our first request to the OpenAI API endpoint. The endpoint that I'll query first is the same endpoint that we hit earlier with curl, the chat completions endpoint, and the way you do this, is by calling client.chat.completions.create. You can see this on line 1. I pass in a value for the model. I want to use the gpt-3.5-turbo model. The second input argument I specify is a list of messages, messages is a Python list, where every entity in the list is a Python dictionary.  
  
[Video description begins] *The Input Cell 4 is is now highlighted. Line 1 reads: response = client.chat.completions.create(. Line 2 reads: model = "gpt-3.5-turbo",. Line 3 reads: messages =[. Line 4 reads: {. Line 5 reads: "role": "system",. Line 6 reads:"content": """. Line 7 reads: As a movie recommender who specializes in various genres,. Line 8 reads: provide information about highly rated and critically acclaimed movies. Line 9 reads:""". Line 10 reads: },. Line 11 reads: {. Line 12 reads: "role": "user",. Line 13 reads:"content": " Can you recommend fun movie for me to watch?". Line 14 reads: }. Line 15 reads:]. Line 16 reads: ). Line 18 reads: response.* [Video description ends]

The first entity on lines 4 through 10 defines the first message object. The role here is system, so these are instructions for the model. I've asked the model to act as a movie recommender who specializes in different genres and gives us recommendations for movies to watch. The second message has the role of the user. This is my prompt. "Can you recommend a fun movie for me to watch?" A very straightforward prompt. client.chat.completions.create will hit the API behind the scenes and the response variable will contain the response.

Just a heads up that this is a synchronous call to the API endpoint, which means we are waiting for the response and then displaying it. If you want this to run asynchronously, you should use the Python async await syntax and I'll show you that in a later demo. Observe that the response is an object of type ChatCompletion with a number of attributes. The response structure is the same as the response structure that we saw in the JSON format when we worked with curl. The actual content of the response is present in the choices member variable.  
  
[Video description begins] *Input Cell 5 reads: print(response.choices[0].message.content).* [Video description ends]  
  
Choices is an array we access the entity at index 0. I access the message within choices and the content of that message, and this is what the response from the model was.

## 5. Video: Exploring Chat Completions and JSON Mode (it\_aiopapdj\_02\_enus\_05)

****

During this video, you will learn how to send prompts to the chat completions API.

* *send prompts to the chat completions API*

[Video description begins] *Topic title: Exploring Chat Completions and JSON Mode. Presented by: Janani Ravi.* [Video description ends]

In this video, we'll continue working with the chat completions API. Let's try another request to this API, and this time, I'll specify multiple messages. Specify multiple messages allows you to set up a conversation when you use an API endpoint. The different messages make up the conversation, and while responding to your latest query, the model will be able to use the context of the previous messages in order to respond.

When you're using the ChatGPT user interface, all of the conversation that you have in a single session, or a single conversation makes up the context that the model has access to. The model then uses that context to respond to your latest query. When we specify multiple messages that we send to an API endpoint, we are essentially simulating the same behavior. Here once again, I call client.chat.completions.create.  
  
[Video description begins] *A tab called BasicPromptsUsing APIs appears underneath Jupyter notebook. The following lines are added in the Input Cell 1. Line 1 reads: response = client.chat.completions.create(. Line 2 reads: model = "gpt-3.5-turbo",. Line 3 reads: messages =[. Line 4 reads: {. Line 5 reads: "role": "system",. Line 6 reads:"content": """. Line 7 reads: As a movie recommender who specializes in various genres,. Line 8 reads: provide information about highly rated and critically acclaimed movies. Line 9 reads:""". Line 10 reads: },. Line 11 reads: {. Line 12 reads: "role": "user",. Line 13 reads:"content": " Can you recommend fun movie for me to watch?". Line 14 reads: },. Line 15 reads: {. Line 16 reads: "role": "assistant",. Line 17 reads:"content": "Only highly recommended and fun movie to watch is 'The Grand Budapest Hotel' (2014)". Line 18 reads: },. Line 19 reads: {. Line 20 reads: "role": "user",. Line 21 reads:"content": "What's the plot of the movie?". Line 22 reads: }. Line 23 reads: ]. Line 24 reads:). Line 26 reads: print(response.choices[0] .message.content).* [Video description ends]  
  
The model is gpt-3.5-turbo, and the messages are a list of messages.

The first message with the role "system" are the instructions that I'm giving the model. The second message with the role "user" is my first query. "Can you recommend a fun movie for me to watch?" The third message with the role "assistant" is a simulated response from the model. Let's say the model says, "One highly recommended and fun movie to watch is 'The Grand Budapest Hotel' (2014)." Now here is a follow-up question that I have in the role "user" "What's the plot of the movie?"

An important thing to note here in the follow-up query is that I do not specify what movie I'm referring to. I expect the model to understand that this is a follow-up query to the previous messages we've had as a part of the same conversation. The model should be able to correctly judge that I'm referencing the plot of The Grand Budapest Hotel. Now, instead of printing out the entire response object, I'm only going to get the choice at index 0, the message, and the content, and here is the response from the model. Notice that the model has correctly identified 'The Grand Budapest Hotel' as the movie that I'm referring to, and it gives me its plot.

Let's say, you're going to be interacting with the API a lot, you may want to set up a utility function that actually talks to the API and you only have to pass in the right arguments to this utility function. This is exactly what I've done here, chat\_with\_model is a utility function that takes in three input arguments, the model\_id, that is the model that I want to interact with, the system\_message that contains the instructions to the model, and the user\_request that contains my prompt.  
  
[Video description begins] *The following lines are added in the Input Cell 2. Line 1 reads: def chat\_with\_model(model\_id, system\_message, user\_message):. Line 2 reads: try:. Line 3 reads: response = client.chat.completions.create(. Line 4 reads: model = model\_id,. Line 5 reads: messages = [. Line 6 reads: {"role": "system", "content": system\_message},. Line 7 reads: {"role": "user", "content": user\_request}. Line 8 reads: ]. Line 9 reads:). Line 11 reads: chat\_response = response.choices[0].message.content. Line 13 reads: return chat\_response. Line 15 reads: except Exception as e:. Line 16 reads: return str(e).* [Video description ends]

Within this function, within a try accept block, I call client.chat.completions.create, pass in the model\_id, as the model that we want to talk to. Set up a list of messages on lines 5 through 8. The first message with the role "system" has the system message as its content. The second message with the role "user" has the user\_request as its content. We extract the chat\_response on line 11 and return it on line 13. If there is an exception while invoking the API on line 16, we return the exception string.

We'll now use this function to interact with the chat completions API. Now the model that I want to talk to is gpt-3.5-turbo, that's my model\_id.  
  
[Video description begins] *Input cell 3 displays the following lines of code. Line 1 reads: model\_id = 'gpt-3.5-turbo'. Line 3 reads: system\_message = """. Line 4 reads: As a movie recommender who specializes in various. Line 5 reads: genres, provide information about highly-rated. Line 6 reads: and critically acclaimed movies in the following format:. Line 7 reads: Title:. Line 8 reads: Director:. Line 9 reads: Release year:. Line 10 reads: Genre:. Line 11 reads: Plot summary:. Line 12 reads: Cast:. Line 13 reads: IMDb rating:. Line 14 reads: Rotten Tomatoes rating:. Line 15 reads: Why I recommend it:. Line 16 reads:""". Line 18 reads: user\_request = " Could you recommend a fun movie for me to watch?". Line 20 reads: response = chat\_with\_model(model\_id, user\_request, system\_message). Line 22 reads print(response).* [Video description ends]  
  
My system\_message sets up the instructions to be very specific. I want the model to be a movie recommender, but I want its responses to have a certain structure. I want the Title of the movie it recommends, the Director, Release year, Cast, IMDb rating, and why the model recommends the movie. My user\_request is the same, "Could you recommend a fun movie for me to watch?" I call chat\_with\_model on line 20. I get the response and print it out the screen and this time around it recommends a different movie. Notice that it follows the structure that I've specified in the instructions.

I get the Title, Director, Release year, bunch of other details, and at the end, why the model chooses to recommend this movie, Why I recommend it. Now in the next example, I'll show you, we'll see how instructions to the model can be very, very specific. We'll continue working with the gpt-3.5-turbo model. On lines 4 and 5, I indicate that the AI assistant generates personalized movie recommendations based on user preferences, and then I give two examples of what inputs and outputs might look like.  
  
[Video description begins] *Input cell 4 displays the following lines of code. Line 1 reads: model\_id = 'gpt-3.5-turbo'. Line 3 reads: system\_message = """. Line 4 reads: You are an AI assistant that generates personalized movie recommendations. Line 5 reads: based on user preferences. Lines 7 to Line 42 contains 2 examples of Input and Output of User profile. Lines 45 to 50 contains prompt with the user details. Line 52 reads: response = chat\_with\_model(model\_id, user\_request, system\_message). Line 54 reads print(response)* [Video description ends]

On lines 9 through 25, I have the first example. The Input will be a User profile defined on lines 9 and 10, Age 25, Preferred genres, Favorite directors, and this is how I want the Output of the model to be defined on lines 12 through 25. And then if you scroll down, you see I have a second similar example for a different user profile defined on lines 27 through 42. Now my prompt defined on lines 45 through 50, just contains a User profile, Age 12, Preferred genres, Animation, Adventure, Fantasy, and Favorite directors. "Can you give me a list of the 3 best movies released in 2018, based on this user's profile?"

Notice how specific I get in the prompt. I call chat\_with\_model on line 52 and I print out the response, and you can see that the response is very relevant and useful. Spider-Man, Incredibles 2, and Mary and the Witch's Flower, these are the three recommendations from my assistant. Next, let's try some zero\_shot prompting using our chat completions API. zero\_shot prompting refers to the capability of natural language models to generate coherent responses for prompts they have never seen during training.  
  
[Video description begins] *The Input Cell highlights the following lines of code. Line 1 reads: def zero\_shot(model\_id, user\_message):. Line 2 reads: try:. Line 3 reads: response = client.chat.completions.create(. Line 4 reads: model = model\_id,. Line 5 reads: messages = [. Line 6 reads: {"role": "user", "content": user\_message}. Line 7 reads: ]. Line 8 reads:). Line 10 reads: chat\_response = response.choices[0].message.content. Line 12 reads: return chat\_response. Line 14 reads: except Exception as e:. Line 15 reads: return str(e).* [Video description ends]  
  
The model is given a prompt or a query for a specific task or question that it hasn't been explicitly trained on. Observe that with zero\_shot, I do not set up a system message, that is, I do not have additional instructions for the model.

The zero\_shot function takes in the model\_id and user\_message. I have the request to the API on lines 3 through 8. Again, this is a synchronous request, it's not running asynchronously. On line 6, I pass in the user\_message with the role "user". On line 10, I extract the content of the response and return that. Now let's use zero\_shot. We'll use the same model, gpt-3.5-turbo. "If a wizard were to recommend a movie to a young apprentice, what movie would they recommend?" So, no additional instructions, that's why this is zero\_shot. But observe, I've specified a constraint, Cartoon Network. This should indicate to the model that the movie should have been on Cartoon Network.  
  
[Video description begins] *The Input cell 11 displays the following lines of code. Line 1 reads: model\_id = "gpt-3.5-turbo". Line 3 reads: user\_message = """. Line 4 reads: If a wizard were to recommend a movie to a young apprentice, what movie would they recommend?. Line 5 reads: constraints: Cartoon Network. Line 6 reads:""". Line 8 reads: response = zero\_shot(model\_id, user\_message). Line 10 reads print(response).* [Video description ends]

On line 8, we call the zero\_shot and then we print out the response. Observe how in the response it uses the constraint and then recommends "The Powerpuff Girls Movie". The response also talks about what the movie might teach the apprentice. Let's try zero\_short once again. The prompt is the same as before, you can see line 4. My constraint is different, I've now set constraints "Disney", I want this to be a Disney movie.

Now here is an example of where the model gets the answer wrong.  
  
[Video description begins] *The Input cell 12 displays the following lines of code. Line 1 reads: model\_id = "gpt-3.5-turbo". Line 3 reads: user\_message = """. Line 4 reads: If a wizard were to recommend a movie to a young apprentice, what movie would they recommend?. Line 5 reads: constraints: Disney. Line 6 reads:""". Line 8 reads: response = zero\_shot(model\_id, user\_message). Line 10 reads print(response).* [Video description ends]  
  
It recommends Harry Potter and the Sorcerer's Stone, but the Harry Potter movie is not a Disney movie, instead, it's part of the Warner Brothers franchise. One thing to remember, while you're working with large language models, always cross-check your responses, the models don't always get things right. Now let's say, you want the responses from the model to be returned in a structured format. A common way to use the chat completions API is to instruct the model to always return a JSON object that makes sense for your use case, and you can specify this as a part of the system message the instructions to the model.   
  
[Video description begins] *The Input cell 13 displays the following lines of code. Line 1 reads: response = client.chat.completions.create(. Line 2 reads: model = "gpt-3.5-turbo-1106",. Line 3 reads: response\_format = { "type ": "json\_object"},. Line 4 reads: messages =[. Line 5 reads: {. Line 6 reads: "role": "system",. Line 7 reads:"content": "You are a helpful assistant designed to output JSON". Line 8 reads: },. Line 9 reads: {. Line 10 reads: "role": "user",. Line 11 reads:"content": " Can you recommend fun movie for me to watch?". Line 12 reads: }. Line 13 reads: ]. Line 14 reads:). Line 16 reads: print(response.choices[0] .message.content).* [Video description ends]  
  
This will cause the model, for the most part, to generate a valid JSON response that you can then parse and use.

Here is how you can use the model in JSON mode. We call client.chat.completions.create, the model is gpt-3.5-turbo-1106. There are only certain models at this point in time that support JSON mode, and the model that I've specified is one amongst these, I pass in an additional response\_format parameter as part of my request, "type": "json\_object". By setting the response format to json\_object, you enable JSON mode and make it more likely that your model will return a valid JSON in the response.

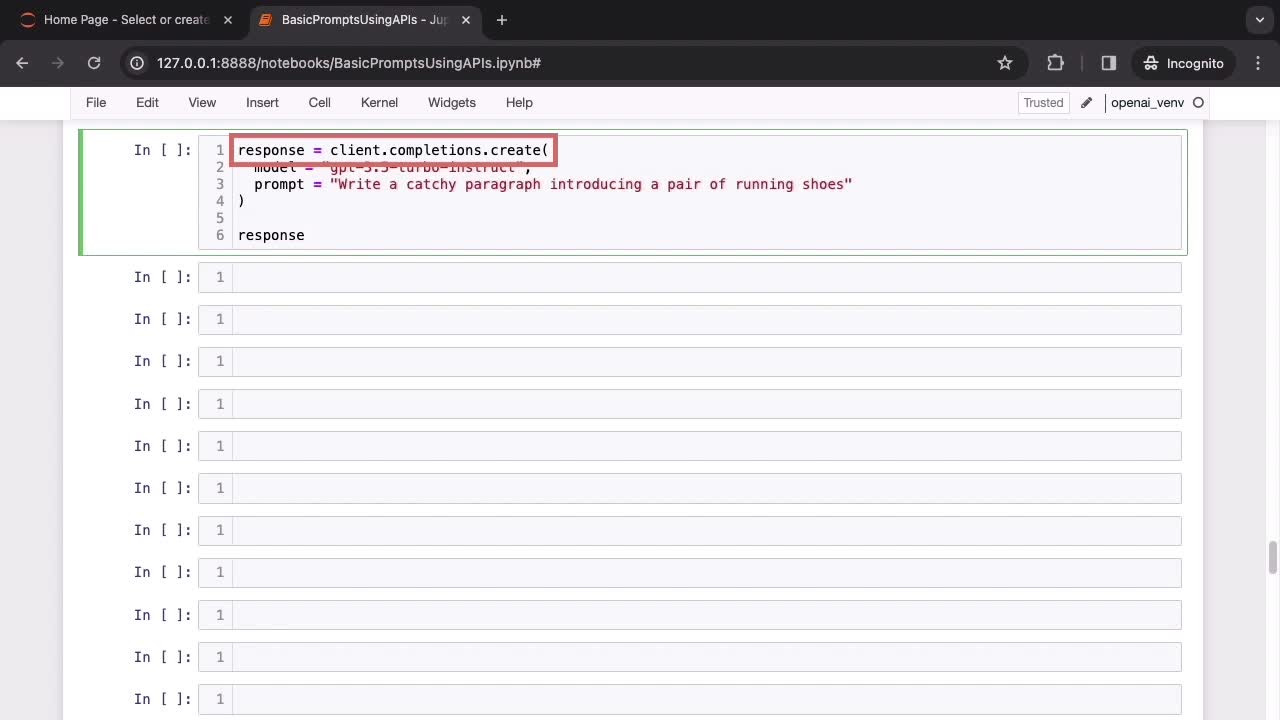
Notice in the system message, I clearly tell the model you're a helpful assistant designed to output JSON. This is important, you have to specify that the output should be JSON as a part of your instructions. If you don't use this word JSON somewhere in your context, the JSON mode will not work. And then as a user I say, "Can you recommend a fun movie for me to watch?" This setup will cause the chat completions API to work in JSON mode and you can see that the response is in a valid JSON format. This is something that you can parse and then use.

A couple of things to remember here. JSON mode does not guarantee what the schema of the resulting JSON will be, only that it will be valid JSON. Sometimes the JSON can be invalid if the response is cut off, let's say if the conversation exceeds the token limit. So, you may not always get JSON, you may not get valid JSON, but you'll mostly get JSON. It's up to you to account for these variations in the response. Let's try this once again. This code is exactly the same as the code that we just saw, with an important difference.

Notice that my system content says, "You're a helpful assistant", It says nothing about JSON.  
  
[Video description begins] *The Input cell 14 displays the following lines of code. Line 1 reads: response = client.chat.completions.create(. Line 2 reads: model = "gpt-3.5-turbo-1106",. Line 3 reads: response\_format = { "type ": "json\_object"},. Line 4 reads: messages =[. Line 5 reads: {. Line 6 reads: "role": "system",. Line 7 reads:"content": "You are a helpful assistant ". Line 8 reads: },. Line 9 reads: {. Line 10 reads: "role": "user",. Line 11 reads:"content": " Can you recommend fun movie for me to watch?". Line 12 reads: }. Line 13 reads: ]. Line 14 reads:). Line 16 reads: print(response.choices[0] .message.content).* [Video description ends]  
  
In fact, in all of the context, there's nothing about JSON. I've set the response format to be of type JSON object, but I don't mention JSON anywhere else. Now, when I run this, this results in an error. It does not know that it has to return JSON unless I mention JSON somewhere in the context of the messages. Let's try JSON mode again, the right way with a slightly different prompt. Remember to set the response\_format as I've done on line 3. Make sure you specify JSON somewhere in the context.

My system content on line 7 says, "You're a helpful assistant designed to output JSON". Remember to use a model that works with JSON mode. gpt-3.5-turbo-1106 does. Here's my query on line 11, "Can you recommend 3 fantasy books that teenagers might love?"  
  
[Video description begins] *The Input cell 13 displays the following lines of code. Line 1 reads: response = client.chat.completions.create(. Line 2 reads: model = "gpt-3.5-turbo-1106",. Line 3 reads: response\_format = { "type ": "json\_object"},. Line 4 reads: messages =[. Line 5 reads: {. Line 6 reads: "role": "system",. Line 7 reads:"content": "You are a helpful assistant designed to output JSON". Line 8 reads: },. Line 9 reads: {. Line 10 reads: "role": "user",. Line 11 reads:"content": " Can you recommend 3 fantasy books that teenagers might love?". Line 12 reads: }. Line 13 reads: ]. Line 14 reads:). Line 16 reads: print(response.choices[0] .message.content).* [Video description ends]  
  
Notice I need 3 fantasy books and you can see that the output here is valid JSON. The books field contains a list of books, each a valid JSON entity.

## 6. Video: Streaming Chat Completions with the Legacy Completions API (it\_aiopapdj\_02\_enus\_06)

****

In this video, find out how to send prompts to the legacy completions API.

* *send prompts to the legacy completions API*

[Video description begins] *Topic title: Streaming Chat Completions with the Legacy Completions API. Presented by: Janani Ravi.* [Video description ends]

If you remember, when we used the APIs from the Open AI Playground, we saw that some models use the legacy completions endpoint. Newer models such as the GPT-3.5 Turbo, GPT-4. These models use the newer chat completions endpoint, and the chat completions endpoint is what we've been working with so far in Python.  
  
[Video description begins] *A tab titled: BasicPromptsUsingAPIs - Jupyter Notebook appears. The main pane contains various menu bar options. Below, it displays various input cells.* [Video description ends]  
  
However, it's possible that you or your organization may need to use the legacy completions endpoint. So for completeness, we'll see how you can use that from Python in this demo. So remember, newer API is chat completions. The legacy API is just legacy completions or just completions.

The legacy completions API received its final update in July 2023 and has a different interface than the new chat completions that we've been using so far. A model that's not deprecated and still supports the legacy completions endpoint is the GPT 3.5 turbo instruct model. So if you want to work with this model, you have to use the legacy API, and the way that's invoked is different. You call client.completions.create rather than client chat completions create. The model I've specified is the gpt-3.5-turbo-instruct model. This won't work with the chat completions API. You have to use the legacy API. Now the legacy API only takes in a prompt as an input. This can be a string or an array of strings. There is no system of messages with rules and conversation. Here my prompt says Write a catchy paragraph introducing a pair of running shoes.

Remember, the instruct model is only good at single question answers. It does not work well with conversation. Let's take a look at the response.  
  
[Video description begins] *Input cell 16 displays the following lines of code. Line 1 reads: response = client.completions.create(. Line 2 reads: model = "gpt-3.5-turbo-instruct",. Line 3 reads: prompt = "Write a catchy paragraph introducing a pair of running shoes". Line 4 reads: ). Line 6 reads: response.* [Video description ends]  
  
Notice the object that's returned in the response is different. It's of type completion. The chat completions API returns an object of type chat completion. Now the actual response of the model can be extracted in this manner. response.choices. This is an array. Let's pick the index position 0 of the array and extract the text member variable.  
  
[Video description begins] *Input cell 17 reads: print(response.choices[0].text).* [Video description ends]  
  
This gives us the actual response. We only receive a partial response because this response was truncated due to length. If you look at the previous result, notice the finish\_reason is length, it's not stop. Stop is when the response is complete and then you get the response. This response was truncated because of length. In the legacy completions API, the max tokens that can be returned in the response is just 16.

And here in the CompletionUsage, the previous response, notice that completion\_tokens is equal to 16. The max tokens was reached and that's why the response was truncated. Let's tweak our call to this API by increasing the max\_tokens that we can receive in the response so that the response is not terminated. I call the legacy API as before client.completions.create. The model is the same, the turbo-instruct model. Just for a little variety my prompt is different. Write a catchy poem talking of how you love SQL databases. You can use the Max tokens parameter to specify the maximum number of tokens to return in the response. I've set this to 128, which is a reasonably good sized response. echo = True will echo back the prompt in addition to the completion.  
  
[Video description begins] *She highlights the following lines of code in Input cell 18. Line 1 reads: response = client.completions.create(. Line 2 reads: model = "gpt-3.5-turbo-instruct",. Line 3 reads: prompt = "Write a catchy poem talking of how you love SQL databases",. Line 4 reads: max\_tokens = 128,. Line 5 reads: echo = True,.* [Video description ends]  
  
The response here is once again a Completion object. The finish\_reason is again length, which means the response will be truncated. Let's extract the text of the response and see what this SQL poem looks like.

The response is much longer because we had set max\_tokens to 128. Let me show you a few other model parameters that you can use to tweak the response from the model using the legacy completions API. My model and prompt are the same as before. I've kept the response short. max\_tokens is equal to 20. I'd like the response to be small because I'm interested in other details of the response. Observe I've set the frequency\_penalty to 1. This is a number that's generally between -2 and 2, and it penalizes new tokens based on their existing frequency in the text so far. So if a token already exists in the text that has been generated, it incurs a penalty. The frequency\_penalty thus decreases the model's likelihood to repeat the same line verbatim. Setting logprobs to 3 includes the log probabilities of the most likely output tokens.

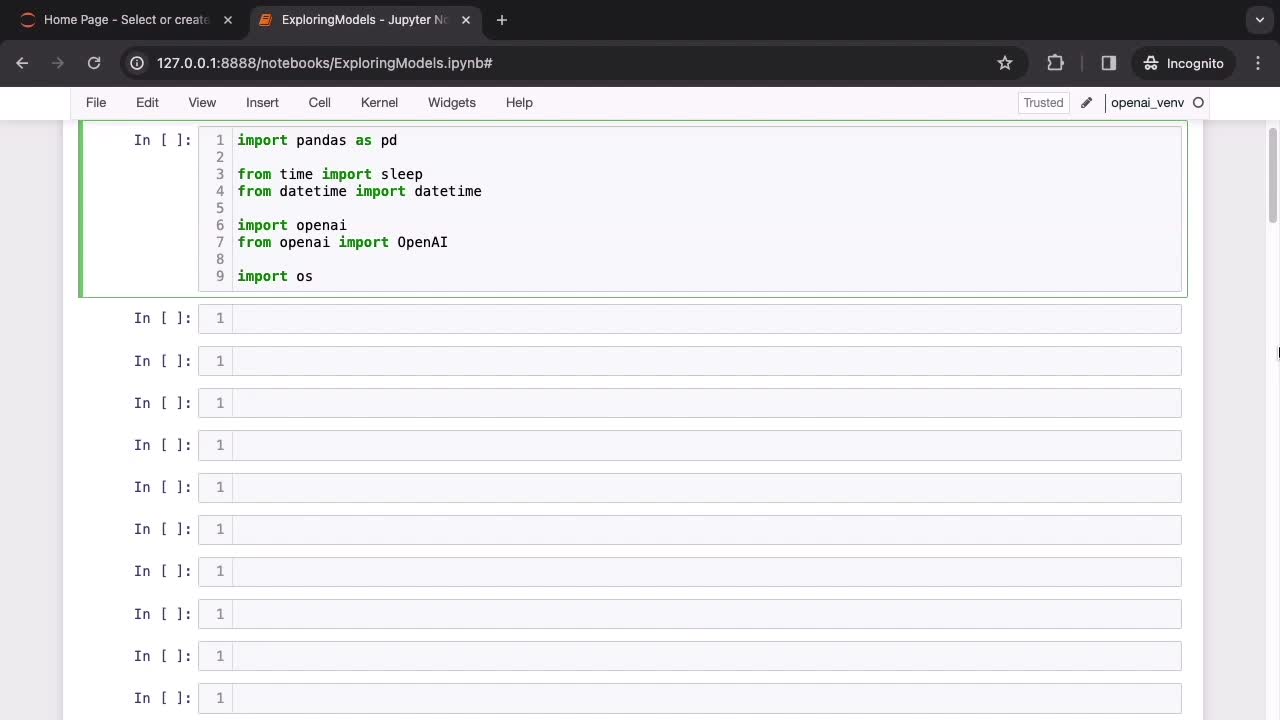
If you remember with the legacy models, we could see the probability scores of the different possible next words in the sequence that could be generated. This is what logprobs will return. A value of 3 here will return the 3 most likely tokens based on their log probabilities.  
  
[Video description begins] *She highlights the following lines of code in Input cell 20. Line 4 reads: max\_tokens = 20,. Line 5 reads: logprobs = 3,. Line 6 reads: frequency\_penalty = 1.* [Video description ends]  
  
Now, the response here is rather a lot, so let's extract portions of the response and take a look. First we look at the actual text of the response. And here you can see that it's a very short poem. Next, in order to view the log probabilities of the various tokens in the response, I'm going to set this up in a Pandas DataFrame. Otherwise it's very hard to see. My DataFrame will contain 2 columns, text and token\_logprobs. For the text I display the text offset, not the actual text, and then I display the log probabilities of each of the selected tokens in the generated sequence.  
  
[Video description begins] *Input cell 22 displays the following lines of code. Line 1 reads: import pandas as pd. Line 3 reads: text\_token\_df = pd.DataFrame(data = {. Line 4 reads: "text": response.choices[0].logprobs.text\_offset,. Line 5 reads: "token\_logprobs": response.choices[0].logprobs.token\_logprobs. Line 6 reads: }). Line 8 reads: text\_token\_df.* [Video description ends]  
  
And this is what the token logprobs look like. If you remember we had set log probs to 3, which means

that for every token in the generated text, we'll have the top 3 tokens that could likely have been selected. This is what I'm trying to view using the JSON PrettyPrinter in the pprint library. So I pprint response.choices[0] . logprobs . top\_logprobs.  
  
[Video description begins] *Input cell 23 displays the following lines of code. Line 1 reads: import pprint. Line 3 reads: pp = pprint.PrettyPrinter(indent=4). Line 5 reads: pp.pprint(response.choices[0].logprobs.top\_logprobs).* [Video description ends]  
  
Here in the response, for every position in the text you can see the 3 tokens that were considered. The first line contains the first set of three tokens, the second line the second set of three tokens. You can see that the first line contains all newline characters. In the second line the 3 possible tokens were I, Oh, or SQL. SQL was the token that was picked. In the third line, you can see the 3 possibilities were databases, is, and the comma. In the generated text, databases was the token that was picked. And so on and so forth for every set of 3 tokens in this list here.

Now I'd like to show you one last feature of the newer chat completions API before we move on to the next demo. The OpenAI API provides the ability to stream responses back to a client in order to allow partial results for requests. Now if you've worked with ChatGPT, you know how the response streams in and then you get like a typewriter like effect on your screen. Receiving partial responses will help reduce the perceived latency for your users. So let's see how you can use streaming support with the chat completions API. Notice we still call client.chat.completions.create. Note once again, this is the newer API. The model I've chosen is gpt-4. The message is just a simple user prompt asking the model to write an e-mail explaining why your project will be delayed, and I've set the input argument stream to be equal to True. Now, by default, when you request a completion from OpenAI, the entire completion is generated before being sent back in a single response. If you're generating long completions, waiting for the response can take many seconds. Stream completions allow you to receive your responses sooner.

You can stream the completion as it's being generated, and this allows you to start printing or processing the beginning of the completion before the full completion is finished. I've stored this response in a stream object and you can now iterate over the stream using a for loop, I do this on line 12. For every chunk in the stream, the partial responses are available in choices index position 0, delta.content. As long as it's not none, let's print out the content.  
  
[Video description begins] *Input cell 24 displays the following lines of code. Line 1 reads: stream = client.chat.completions.create(. Line 2 reads: model = "gpt-4",. Line 3 reads: messages=[. Line 4 reads: {. Line 5 reads: "role": "user",. Line 6 reads: "content": "write an email explaining to your boss why your project will be delayed". Line 7 reads: }. Line 8 reads: ],. Line 9 reads: stream=True,. Line 10 reads: ). Line 12 reads: for chunk in stream:. Line 13 reads: if chunk.choices[0].delta.content is not None:. Line 14 reads: print(chunk.choices[0].delta.content, end="").* [Video description ends]  
  
I'm going to go ahead and hit Enter and in a few moments you'll see how the responses stream in. Observe that I don't get the entire response right away, but instead portions of the response come in and are printed out to screen, thanks to my print statement. All you have to do is set stream to True and then iterate over the chunks that you receive to get a streaming response.

## 7. Video: Exploring Chat and Completion Models (it\_aiopapdj\_02\_enus\_07)

****

In this video, you will learn how to send prompts to different models and view the results.

* *send prompts to different models and view the results*

[Video description begins] *Topic title: Exploring Chat and Completion Models. Presented by: Janani Ravi.* [Video description ends]

Here I am in a brand new notebook for our next demo. Now so far, when we've been working with our chat completion APIs, we've been primarily using the GPT-3.5 Turbo or the GPT-4 model. But early on in this course, we saw that OpenAI has many different models for the different use cases with different pricing as well. This demo, the name of the notebook is called ExploringModels, and we'll explore a few different models that OpenAI has to offer.  
  
[Video description begins] *A tab titled: ExploringModels - Jupyter Notebook appears. The main pane contains various menu bar options. Below, it displays various input cells.* [Video description ends]  
  
I should give you a little bit of a heads up here, though. Over the course of the year 2023, OpenAI released several different versions of different models, and it released GPT-4, which had abilities beyond GPT-3.5. So you may find that when you're watching this course, there are many new models that OpenAI has to offer. As we explore the different models in this demo, my recommendation would be that you open up this particular URL platform.openai.com/docs/models.

This is the URL which will give you the latest and greatest models that are available on Open AI and what the different models are used for. You might see that certain models that I'm using were the latest at the time of my recording, but are not the latest anymore. This is the URL where you can find the newest releases, so I recommend that you spend some time on this page while doing this demo. Let's set up the imports for all of the different libraries that we plan to use. I have a few utility libraries here as well. pandas, sleep, datetime. Make sure you import openai from OpenAI and import os. We'll use the os library to set up the environment variable with our  
  
[Video description begins] *Input cell 1 displays the following lines of code. Line 1 reads: import pandas as pd. Line 3 reads: from time import sleep. Line 4 reads: from datetime import datetime. Line 6 reads: import openai. Line 7 reads; from openai import OpenAI. Line 9 reads: import os.* [Video description ends]  
  
OPENAI\_API\_KEY, and then we'll instantiate the OpenAI client.  
  
[Video description begins] *Input cell 2 displays the following lines of code. Line 1 reads: os.environ["OPENAI\_API\_KEY"] = "sk-n8ycz62S. The rest of the code displays a similar pattern of characters. Line 3 reads: client = OpenAI().* [Video description ends]

You'll find that this is boilerplate code that I set up at the start of every demo. If you want to see a list of all of the models that OpenAI currently offers, you can call client.models.list. This will hit the REST endpoint where you can retrieve all of the models.  
  
[Video description begins] *Input cell 3 displays the following lines of code. Line 1 reads: models = client.models.list(). Line 3 reads: models.* [Video description ends]  
  
We saw that using curl and it'll give you a response. And here you can see rather hard to read all of the models that OpenAI has to offer for different use cases. There's a lot of extraneous information here in that response. Let's only look at the models that's available in models.data and here you can see a list of models. Every entity in the list is a Model object. Every model has an ID, a created date, an object which is of type model, and who the model is owned by, usually openai or openai-dev. Some models are owned by system as well. Responses of this type can be hard to interpret and understand, so it's a good practice to format the response into a form that is easy for you to interpret. Here I initialize a Python list called models\_list and then I run a for loop over the data returned by the API response.

Then for every model in that list, I construct a dictionary. The dictionary on lines 4 through 9 extracts the model's id, the model's object attribute, and it converts the created field to be of type datetime so that it's human-readable. We extract who the model is owned\_by and then we append this model dictionary to the model list. On line 12, I construct a DataFrame from the list of dictionary objects and I sample 15 records from that DataFrame.  
  
[Video description begins] *Input cell 5 displays the following lines of code. Line 1 reads: models\_list = []. Line 3 reads: for model in models.data:. Line 4 reads: model\_dict = {. Line 5 reads: "id": model.id,. Line 6 reads: "object": model.object,. Line 7 reads: "created": datetime.utcfromtimestamp(model.created).strftime('%Y-%m-%d %H:%M:%S'),. Line 8 reads: "owned\_by": model.owned\_by,. Line 9 reads: }. Line 10 reads: models\_list.append(model\_dict). Line 12 reads: df = pd.DataFrame(models\_list). Line 14 reads: df.sample(15).* [Video description ends]  
  
And here is a sample of 15 models. Notice that most of these models have been created in the year 2022 or 2023. There are two models here that are fine-tuned models, the one that index 68 and 69. In order to explore responses from different models, I'm going to set up a little helper function here called generate\_chat\_response. It takes in the model\_id of the model I want to use.

It'll take in a user\_message and we can take in max\_tokens, which by default is 256. I call client.chat.completions.create to make a request to the chat completions endpoint, specify the model\_id on line 4, pass in the messages on line 5 through 8. I've also specified values for some other parameters for the model response. I'm going to set the temperature to 0 so the model works at its most deterministic and max\_tokens is set to the max tokens that we pass in. I'm going to print out the model that was used to produce the response as well as the content of the model's response. This I do on lines 12 and 13.  
  
[Video description begins] *She highlights the following lines of code in Input cell 6. Line 1 reads: def generate\_chat\_response(model\_id, user\_message, max\_tokens=256):. Line 2 reads: try:. Line 3 reads: response = client.chat.completions.create(. Line 4 reads: model = model\_id,. Line 5 reads: messages = [. Line 6 reads: {"role": "system", "content": "You are a helpful assistant."},. Line 7 reads: {"role": "user", "content": user\_message}. Line 8 reads: ],. Line 9 reads: temperature = 0,. Line 10 reads: max\_tokens = max\_tokens,. Line 11 reads: ). Line 12 reads: print(response.model). Line 13 reads: print(response.choices[0].message.content).* [Video description ends]  
  
Now that I have my helper function set up, let's explore some of the different models that OpenAI has to offer. Now, the first model I've picked is one we've used before. The gpt-3.5-turbo, refers to a recent release of the GPT-3.5 Turbo model.

The actual release is not specified. I ask the model to perform a simple translation from Spanish to English and I've specified the text and I invoke generate\_chat\_response, pass in the model\_id and the user\_message. And you can see that the gpt-3.5-turbo-0613 model is actually what performs the translation.   
  
[Video description begins] *Input cell 7 displays the following lines of code. Line 1 reads: model\_id = 'gpt-3.5-turbo'. Line 2 reads: user\_message = "Translate the following Spanish text to English: 'El perro corre en el parque.". Line 4 reads: generate\_chat\_response(model\_id, user\_message).* [Video description ends]  
  
This is the specific version of the model along with the release date. Now let's try the same translation, but this time with the gpt-4 model. gpt-4 here is not the precise model\_id, It references one of the latest releases of the gpt-4 model.  
  
[Video description begins] *She highlights the code in Input cell 8 that reads: model\_id = 'gpt-4'.* [Video description ends]  
  
When you look at the response, you can see that the actual model that was used was gpt-4-0613.

It's possible that when you are running this code, a newer version of the gpt-4 model will be used for the translation. Now let's say you specify a model that does not exist like I've done here, gpt-4-32k-0613, and I've asked you to perform translation. Now this is a model that does not exist and this will give you an Error message. The message clearly says that The model does not exist or we don't have access to it. Now, we know that we do have access to GPT-4 models, so likely this particular model with this ID does not exist. Now if you want to use a specific version of a model, it's possible to specify that as a model\_id as well. Here I want to use gpt-4-0613, not any other version of gpt-4, but this specific version, and this is the version that I get to perform the translation. GPT-3.5 and GPT-4 are the latest models. Next we'll see how we can work with legacy models. Now, I should warn you that some of the legacy models that I'm using now may be deprecated by the time you are watching this video.

If that's indeed the case, my suggestion would be to head over to the Models page, find some legacy models that have not been deprecated, and replace those model ids instead of the ones that I've chosen. Now here I've chosen the text-davinci-003 legacy model. The legacy models use the legacy completions API. If you try to use them with chat completions, you'll get an Error message. This is not a chat model and thus not supported in the v1/chat/completions endpoint. Did you mean to use v1 completions? That is the legacy endpoint. Now I'm going to update my utility function so that it can query both the chat endpoint and the legacy completions endpoint. The utility function is called generate\_chat\_response. It takes in the model\_id, the user\_message, max\_tokens is set to 256. If the model\_id starts with gpt-, you can see that I use the chat completions endpoint, client.chat.completions.create. We send the request using the new chat completions endpoint.  
  
[Video description begins] *She highlights the following lines of code in Input cell 12. Line 1 reads: def generate\_chat\_response(model\_id, user\_message, max\_tokens=256):. Line 2 reads: try:. Line 3 reads: if model\_id.startwith("gpt-"):. Line 4 reads: response = client.chat.completions.create(. Line 5 reads: model = model\_id,. Line 6 reads: messages = [. Line 7 reads: {"role": "system", "content": "You are a helpful assistant."},. Line 8 reads: {"role": "user", "content": user\_message}. Line 9 reads: ],. Line 10 reads: temperature = 0,. Line 11 reads: max\_tokens = max\_tokens,. Line 12 reads: ). Line 13 reads: print(response.model). Line 14 reads: print(response.choices[0].message.content).* [Video description ends]

If the model\_id starts with something other than gpt-, then the code on lines 16 through 24 will be executed.  
  
[Video description begins] *Line 16 reads: response = client.completions.create(. Line 17 reads: model = model\_id,. Line 18 reads: prompt = user\_message,. Line 19 reads: temeperature = 0,. Line 20 reads: max\_tokens = max\_tokens. Line 21 reads: ). Line 23 reads: print(response.model). Line 24 reads: print(response.choices[0].text).* [Video description ends]  
  
Here notice that I use client.completions.create on line 16. That is the legacy endpoint. So I'm just treating models differently. Now on line 3, if model\_id starts with gpt-, that's just a rough rule of thumb that I've set up. OpenAI may have newer models that don't start with gpt- and use the chat completions endpoint. It's just a rough rule of thumb that I've come up with that newer chat models tend to start with the prefix gpt-. There is one model for which this if condition won't work. The GPT 3.5 Turbo instruct model starts with GPT, but it uses the legacy completions API. Just something for you to watch out for.

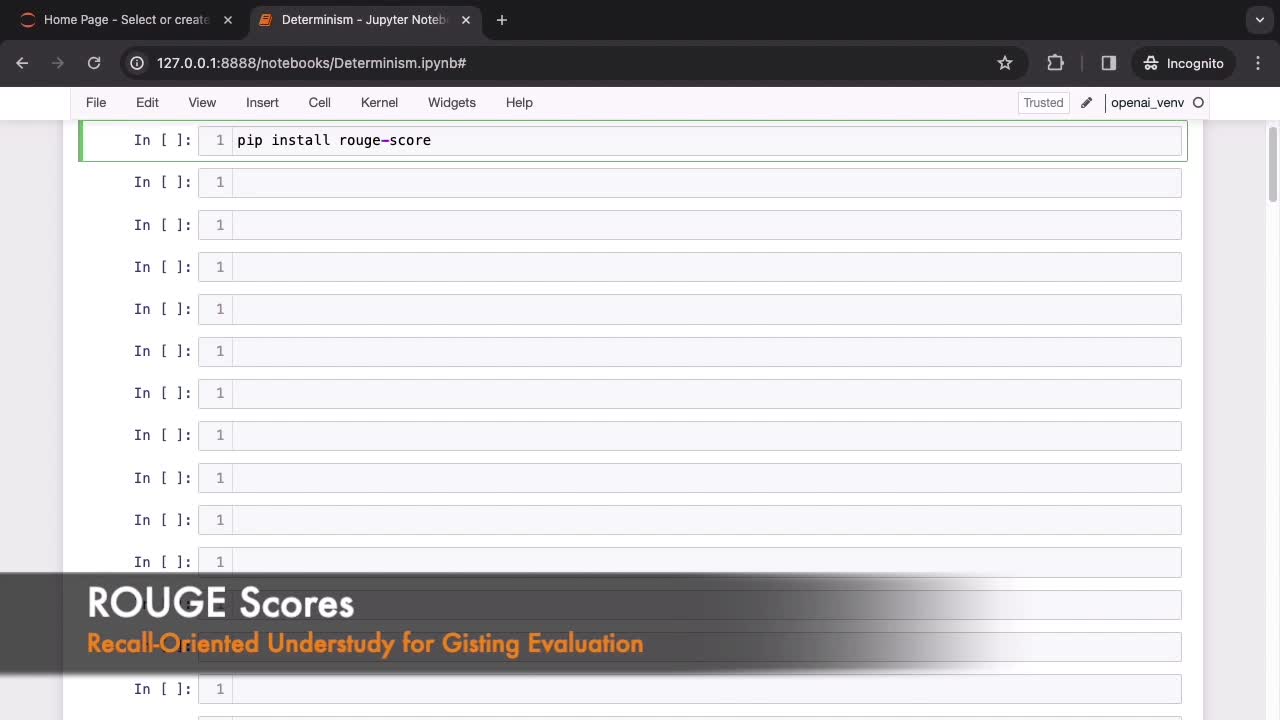
So let's use the older model text-davinci-003 and ask it to perform a translation. text-davinci-003 is a fairly capable model and can perform translations. Just a heads up that if text-davinci-003 is deprecated, use gpt-3.5-turbo instruct that has capabilities similar to text-davinci-003. If you're using the GPT 3.5 Instruct instead of text-davinci-003, remember that it uses the legacy completions endpoint. You may have to update our generate\_chat\_response code. What I'm going to do now is set up a bunch of different OpenAI models in a list, and we'll query each of the models with the same prompt. Other than the last two models here, gpt-3.5 and gpt-4, the remaining models are all legacy models. You'll see that many of the legacy models won't perform well at all on tasks that we take for granted with GPT-3.5 and GPT-4.  
  
[Video description begins] *Input cell 14 displays the following lines of code. Line 1 reads: model\_ids = [. Line 2 reads: "babbage-002",. Line 3 reads: "text-babbage-001",. Line 4 reads: "davinci",. Line 5 reads: "curie-instruct-beta",. Line 6 reads: "curie",. Line 7 reads: "davinci-instruct-beta",. Line 8 reads: "text-curie-001",. Line 9 reads: "text-davinci-001",. Line 10 reads: "text-davinci-002",. Line 11 reads: "text-davinci-003",. Line 12 reads: "ada",. Line 13 reads: "text-ada-001",. Line 14 reads: "gpt-3.5-turbo",. Line 15 reads: "gpt-4". Line 16 reads: ].* [Video description ends]  
  
The first task that I set up is to ask the model to write a Python function that calculates the factorial of a given integer.

On line 5, I iterate over all of the model ids and on line 7, I pass in the model\_id to generate chat response to get the response from the model, and then I sleep for a bit so as to not overwhelm the API endpoint.  
  
[Video description begins] *Input cell 15 displays the following lines of code. Line 1 reads: user\_message = "Write a Python function that calculates the factorial of a given integer.". Line 3 reads: separator = "=" \* 80. Line 5 reads: for model\_id in model\_ids:. Line 6 reads: print(separator). Line 7 reads: generate\_chat\_response(model\_id, user\_message). Line 8 reads: print(separator). Line 9 reads: sleep(15).* [Video description ends]  
  
Let's go ahead and see how these models do. You can see the babbage-002 model just reflects the prompt back at me. The text-babbage-001 is able to compute the factorial. The davinci model seems to produce code, but not all of the code that it has produced is correct. If you look further, the older models, the curie models really can't produce code at all. As you scroll through the results, you can see that the older models can't perform this task at all. But as we get to the newer models, such as the text-davinci models, the things seem to get better.

Notice the newest models, the gpt-3.5-turbo that can produce perfect code, as can gpt-4. Let's try one last prompt across all of these models. Translate the phrase, I love learning new languages into Hindi. Now, once again, we'll try the entire gamut of models. You can see that babbage-002 can't do this at all. And here, none of the other models seem to know any Hindi. They're just producing any kind of junk. The curie-instruct-beta model produces Hindi, but it's not actually the right translation. The curie model can't really translate stuff. The text-davinci-001 model produces a poor translation, but davinci-002, and davinci-003 have pretty good translations. Let's scroll further down. The ada models are old models. They can't perform translation. And of course, gpt-3.5-turbo and gpt-4,

they do very well. Wait, I thought of one more prompt I'd like to try across all of these models. Let's see creative writing , Write a creative and engaging short story about a detective solving a mysterious murder case in a small town. Let's see how these models do. You can see the babbage model is a much older model. It can't really do any creative writing. The text-babbage model tries, but it's not that interesting and you can scroll further and take a look. The text-davinci models are good at writing tasks and all of them produce pretty good results. And of course, the latest models, the gpt-3.5-turbo and gpt-4, both do very well in this kind of creative writing. They produce interesting stories.

## 8. Video: Setting up Utility Functions to Explore Model Determinism (it\_aiopapdj\_02\_enus\_08)

****

During this video, discover how to configure functions to check determinism.

* *configure functions to check determinism*

[Video description begins] *Topic title: Setting up Utility Functions to Explore Model Determinism. Presented by: Janani Ravi.* [Video description ends]

Here I am in a brand new notebook for a brand new demo on determinism. You can see the name of the notebook in the URL Determinism.ipynb#  
  
[Video description begins] *A tab titled: Determinism - Jupyter Notebook appears. The main pane contains various menu bar options. Below, it displays various input cells.* [Video description ends]  
  
Now, when you're integrating OpenAI's models into your applications, there is one drawback that you always face. Modeled responses are not deterministic in nature, so you cannot accurately predict how a model will respond to your query.

You may be certain that the model is giving you the correct response, but it's hard for you to fathom the exact format of the response or the exact contents of the response. This might make it hard for you to use these generative models in certain kinds of applications, so it shouldn't be surprising to you that determinism has always been a big request from the users of these models.

If it were possible to have the model's response always be along certain lines, well that can be a huge plus when integrating generative AI tools. Now, with some recent versions of OpenAI models, it's possible for you to get mostly deterministic responses using the seed parameter. Please note the use of the term mostly deterministic. The responses can never be fully deterministic. In order to be able to compare model responses to see how different or similar they are, I'm going to install a library known as rouge-score.  
  
[Video description begins] *Input cell 1 reads: pip install rouge-score.* [Video description ends]

ROUGE here is an acronym which stands for Recall-Oriented Understudy for Gisting Evaluation. This is a metric usually used to evaluate the quality of machine-generated text. ROUGE Scores are usually used to assess the similarity between a reference human-generated summary and a system or a machine-generated summary. The main idea behind ROUGE is to measure how well the generated text aligns with the reference text in terms of content overlap.

Thus, ROUGE metrics assess the recall or the ability of the generated text to capture all of the important information from the reference text. We'll be using ROUGE metrics in a slightly different way. We don't have any reference human-generated text. Instead, we'll compare the text generated by two different models to see how similar those texts are. This will allow us to see the effect of model determinism using the seed parameter.

This rouge-score library will allow us to compute rouge-score, so go ahead and install it. Once the installation is complete, let's go ahead and set up the imports for the various libraries that we'll be using. Now, of course, we need OpenAI. I also import a library called difflib that will allow us to diff the contents of text generated by two different models. From IPython.display, I import display and HTML functions that will allow us to display HTML formatted output within our notebook.

And from rouge\_score we import the rouge\_scorer, allowing us to calculate rouge metrics.  
  
[Video description begins] *She highlights the following lines of code in Input cell 2. Line 8 reads: import difflib. Line 9 reads: from IPython.display import display, HTML. Line 11 reads: from rouge\_score import rouge\_scorer.* [Video description ends]  
  
Then we have the usual boilerplate code setting up the "OPENAI\_API\_KEY" environment variable and instantiating the OpenAI client.  
  
[Video description begins] *Input cell 3 displays the following lines of code. Line 1 reads: os.environ["OPENAI\_API\_KEY"] = 'sk-n8ycz62S. The rest of the code displays a similar pattern of characters. Line 3 reads: client = OpenAI().* [Video description ends]  
  
The chat completion API is non-deterministic by default, which means that the output from the model may differ from request to request even if the request is exactly the same. However, OpenAI now offers some control over generating deterministic outputs for your prompts.

There are a few model level controls that you can use to make sure that your outputs are similar, if not exactly the same. I'm going to set up a helper function that'll allow us to query the chat completions API. The function is called get\_chat\_response. Notice that I mark this function as async. The async keyword indicates to Python that the execution of this function should be performed asynchronously on a parallel thread.

That is, the main line of execution should not be blocked while this function executes. The get\_chat\_response function takes in a model\_id and a user\_message and a seed parameter that is set to None by default. When seed is set to None, the model responses will be non-deterministic. Next, we call the chart completions API client.chat.completions.create on line 3. We pass in the model\_id on line 4, the messages on lines 5 through 8.

I'll set the temperature parameter to be the same across all our model requests. This is important if you want to get deterministic responses. All of the other parameters to the model should be exactly the same across requests. I'm always going to be using a temperature of 0.5. If you want even more determinism, you may want the temperature parameter to be 0. The seed parameter comes from the input argument seed, and I've set max\_tokens to be 256. That will control the length of the response.  
  
[Video description begins] *She highlights the following lines of code in Input cell 4. Line 1 reads: async def get\_chat\_response(model\_id, user\_message, seed = None):. Line 2 reads: try:. Line 3 reads: response = client.chat.completions.create(. Line 4 reads: model = model\_id,. Line 5 reads: messages = [. Line 6 reads: {"role": "system", "content": "You are a helpful assistant."},. Line 7 reads: {"role": "user", "content": user\_message}. Line 8 reads: ],. Line 9 reads: temperature = 0.5,. Line 10 reads: seed = seed,. Line 11 reads: max\_tokens = 256.* [Video description ends]

The only parameters that we can change will be the prompt that we sent to the model. The model that we use, that is the model\_id and the seed. The seed is what we'll use to control determinism. The other model parameters such as temperature and max\_tokens and other values that we have not specified and which take on their default values. Those will remain the same across all API calls. The seed parameter is what is used to control determinism.

You can set the seed value to any integer of your choice, but for deterministic results you have to set the same seed value across requests. With the same value of seed for the same model, the response that you get will be mostly deterministic. It will not be a 100% deterministic, that's just something to watch out for. Once we get the response from the model, we extract the response\_content on line 14 and we access the system\_fingerprint on line 15.

The system\_fingerprint is an identifier for the current combination of model weights, infrastructure, and other configuration options used by OpenAI servers to generate the completion. The system\_fingerprint will change if you change any of the request parameters or when OpenAI updates the numerical configuration of the infrastructure serving its models. So, the system\_fingerprint kind of uniquely identifies the parameters you've used to make the request and OpenAI's configuration for that particular model that served the response.

So, if the system\_fingerprint changes, you know that either you've changed your request parameters or OpenAI has changed something and that's why the response is no longer mostly deterministic. So if the seed, the request parameters, and system\_fingerprint all match across your request, then model outputs will mostly be identical. There is a small chance that responses differ even when request parameters and system\_fingerprint match due to the inherent non-determinism of how models work.

Next, on line 16 we extract the prompt\_tokens, that is the number of tokens in the prompt, and then on line 17, we compute the completion\_tokens, that is total\_tokens - prompt\_tokens.  
  
[Video description begins] *Line 14 reads: response\_content = response.choices[0].message.content. Line 15 reads: system\_fingerprint = response.system\_fingerprint. Line 16 reads: prompt\_tokens = response.usage.prompt\_tokens. Line 17 reads: completion\_tokens = (. Line 18 reads: response.usage.total\_tokens - response.usage.prompt\_tokens. Line 19 reads: ).* [Video description ends]  
  
On lines 21 through 28, we construct an HTML table with all of the information that we've extracted from the model's response, the Response, the Fingerprint, the Number of prompt tokens, Number of completion tokens. And then we display this (HTML (table)) on line 29 and also return the response\_content on line 31  
  
[Video description begins] *Line 29 reads: display(HTML(table)). Line 31 reads: return response\_content.* [Video description ends]

I have my helper function for generating the model response. Next, I set up another helper function to compare\_responses. This takes in two input arguments, previous\_response and response, and we use the differ from difflib to compare these responses. On line 5, I split both the previous\_response and the current response on the period. So, I get the individual sentences in the response and then I call d.compare to see how the individual sentences differ and I place the result in the diff variable.  
  
[Video description begins] *She highlights the following lines of code in Input cell 5. Line 1 reads: def compare\_responses(previous\_response, response):. Line 3 reads: d = difflib.Differ(). Line 5 reads: diff = d.compare(previous\_response.split("."), response.split(".")).* [Video description ends]

We then set up a table with the differences and similarities. In the HTML table, for every line in diff that's the for loop on line 10. If the line starts with ("- "), we know it's a line that was present in the previous\_response but is not present in the current response, and we'll display that in red. That's on line 12. Else, if the line starts with ("+ "), then we know it's a line that's present in the current response but was not there in the previous\_response, and we'll display that in green.

This is the code on line 15. Otherwise, both the lines match and we'll just display that line in black. That's the code on line 18.  
  
[Video description begins] *Line 10 reads: for line in diff:. Line 11 reads: if line.startswith("-"):. Line 12 reads: diff\_table += f"<tr style='color: red;'><td>{line}</td></tr>". Line 14 reads: elif line.startswith("+"):. Line 15 reads: diff\_table += f"<tr style='color: green;'><td>{line}</td></tr>". Line 18 reads: diff\_table += f"<tr><td>{line}</td></tr>".* [Video description ends]  
  
On lines 22 through 25, if a difference exists, we'll display the table of differences. Otherwise, we'll print ("No differences found.") Next, we instantiate the ROUGE scorer that will allow us to compare model responses to see how closely they overlap in text.  
  
[Video description begins] *Line 22 reads: if diff\_exists:. Line 23 reads: display(HTML(diff\_table)). Line 24 reads: else:. Line 25 reads: print("No differences found.").* [Video description ends]

Now there are several different ROUGE metrics that can be computed for the different bits of text. Here I'm computing 3 different ROUGE metrics, 'rouge1', 'rouge2' and 'rougeL'. I'll explain what those are in just a bit. I've also set use\_stemmer = True so that the individual words in the text are converted to their root forms before comparison. In rouge1 and rouge2, the terms 1 and 2 refer to n-grams, that is, unigrams and bigrams respectively.

rouge1 will measure the overlap of unigrams or single words between the reference and the actual text. rouge2 will measure the overlap of bigrams between the reference and the comparison text. rougeL calculates the longest common subsequence between the two texts, and it represents the longer sequence of words that appears in both the reference and the comparison text.

Once we instantiate the rouge\_scorer, I'll use this to compute Rouge metrics on a few different pairs of texts so that you understand how the rouge\_score works.   
  
[Video description begins] *Input cell 6 displays the following lines of code. Line 2 reads: scorer = rouge\_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use\_stemmer = True). Line 4 reads: scorer.* [Video description ends]  
  
On lines 2 and 3, I've defined the reference\_text and the predicted\_text and you can see that they are exactly the same. Next on line 6, I pass in this reference and predicted\_text into the rouge\_scorer. This will compute 3 different metrics for rouge1, rouge2, and rougeL respectively.

The three metrics are Precision, Recall, and F1 Score. The precision for each of the ROUGE metrics measures the ratio of the number of overlapping n-grams between the reference\_text and the predicted\_text to the total number of n-grams in the predicted\_text. Recall measures the ratio of the number of overlapping n-grams between the predicted\_text and the reference\_text to the number of n-grams in the reference\_text.

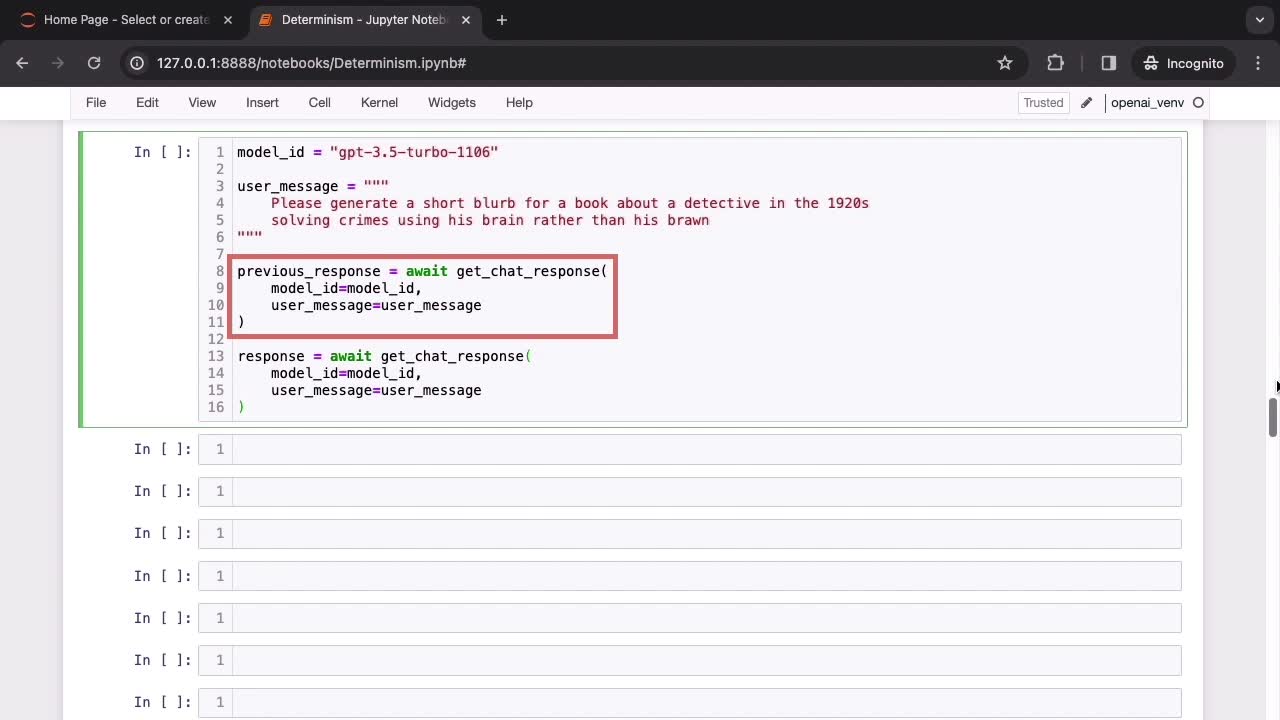
And finally, the F1 Score is a combination of Precision and Recall, which provides a balanced measure of how well the predicted\_text matches the reference\_text. We'll mostly just focus on this F1 Score rather than look at individual Precision, Recall metrics.  
  
[Video description begins] *Input cell 7 displays the following lines of code. Line 2 reads: reference\_text = "This book is thrilling and will keep you gripped till the very end". Line 3 reads: predicted\_text = "This book is thrilling and will keep you gripped till the very end". Line 6 reads: rouge\_scores = scorer.score(reference\_text, predicted\_text). Line 9 reads: for metric, scores in rouge\_scores.items():. Line 10 reads: print(f'{metric}:'). Line 11 reads: print(f' Precision: {scores.precision}'). Line 12 reads: print(f' Recall: {scores.recall}'). Line 13 reads: print(f' F1 Score: {scores.fmeasure}').* [Video description ends]  
  
Let's compute the ROUGE metrics for these identical texts. Notice that the Precision, Recall, and F1 Score are all equal to 1.0. The two texts match perfectly across all ROUGE metrics, rouge1, 2 and rougeL

Now let's make the predicted\_text, which we'll consider the second response from the model, a little difference from the reference\_text. You can see that the predicted\_text has a few additional words and a few words that are different. Once again, we'll compare the ROUGE score between the two and we'll print out Precision, Recall, and F1 Score. The rouge1 F1 Score is 0.78, so below 1, the rouge2 F1 Score is even lower, 0.61, and the rougeL F1 Score is 0.71.

The two texts here are a little different, though largely similar. Let's try this again. I'm going to make the predicted\_text very different from the reference\_text. The reference text talks about the book being thrilling. Predicted text uses completely different words. It talks about the book being boring. Let's take a look at the computed ROUGE scores here and you can see that they are very low.

The F1 Scores are all below 0.5, the rouge1 F1 Score is just 0.36, rouge2 is just 0.1, and the rougeL F1 Score is 0.36. You can see that the two texts that we have compared are very different indeed. We now have everything set up to see how determinism works with chat completion models. We'll do that in the next video.

## 9. Video: Using the Seed Parameter for Almost Deterministic Responses (it\_aiopapdj\_02\_enus\_09)

****

Find out how to adjust the seed parameter.

* *adjust the seed parameter*

[Video description begins] *Topic title: Using the Seed Parameter for Almost Deterministic Responses. Presented by: Janani Ravi.* [Video description ends]

With our utility function setup, we are now ready to see how model determinism works. Now determinism is a fairly new feature at the time of this recording, and you'll find that the seed parameter only works with newer models that support determinism.  
  
[Video description begins] *A tab titled: Determinism - Jupyter Notebook appears. The main pane contains various menu bar options. Below, it displays various input cells.* [Video description ends]  
  
Now in this first example here, I'll generate responses from the same model, but I won't specify the seed parameter. And you'll see that the model responses will be very different.

Now, the model that I have selected is the "gpt-3.5-turbo-1106". What I've noticed with model determinism is that you have to pick a model explicitly and specify the version. You don't get a fingerprint, and determinism will not work if you just pick, say, gpt-3.5-turbo. That's because that is likely to pick the latest gpt-3.5-turbo model, which can be any version based on when you're actually using the API.

Notice that I've explicitly specified that I want gpt-3.5-turbo version 1106. Next, I specify my prompt in the user\_message.  
  
[Video description begins] *She highlights the following lines of code in Input cell 10. Line 1 reads: model\_id = "gpt-3.5-turbo-1106".* [Video description ends]  
  
Please generate a short blurb for a book about a detective in the 1920s solving crimes using his brain rather than his brawn.  
  
[Video description begins] *Line 3 reads: user\_message = """. Line 4 reads: Please generate a short blurb for a book about a detective in the 1920s. Line 5 reads: Solving crimes using his brain rather than his brawn. Line 6 reads: """.* [Video description ends]  
  
Now we'll generate the response from the models using get\_chat\_response. Remember that it's an async function so you'll need to invoke it using the await keyword.

The await keyword indicates to Python that we are running an asynchronous operation, and it should pause and allow other tasks to complete till the awaited operation is completed. So on lines 8 through 11, we make our first request to the model to get a response to our prompt and I save that in previous\_response and then I make another request to the same model with the same prompt on lines 13 through 16.

I haven't specified the seed parameter so seed is set to None by default, which means that the model's response will not be deterministic. You can expect that there will be large variations in the response.  
  
[Video description begins] *Line 8 reads: previous\_response = await get\_chat\_response(. Line 9 reads: model\_id=model\_id,. Line 10 reads: user\_message=user\_message. Line 11 reads: ). Line 13 reads: response = await get\_chat\_response(. Line 14 reads: model\_id=model\_id,. Line 15 reads: user\_message=user\_message. Line 16 reads: ).* [Video description ends]  
  
Let's go ahead and get the two responses from the models and compare them. You can see that we've output the model responses in the form of an HTML table that makes it easier for us to read and parse. And if you look at the two responses you can see at the glance, they are very different. The first Response has 126 tokens in the Response and the second Response is shorter with just 87 tokens.

Observe that the System Fingerprint of the model is the same, it starts with fp and ends with bb. Remember the System Fingerprint includes the request parameters, the seed and the model configuration on the OpenAPI server. Now if you remember we have a utility function that allows us to compare these two responses. I'm going to invoke compare\_responses, pass in previous\_response and current response and we'll get a line-by-line comparison.

The lines that you see in green are the new lines that  
  
[Video description begins] *Input cell 11 reads: compare\_responses(previous\_response, response).* [Video description ends]  
  
are present in the second Response but not in the first. The lines that you see in red are the lines that are present in the first Response but not in the second. Another way to compare the two responses is to compute the rouge\_score between the current response and the previous\_response,  
  
[Video description begins] *Input cell 12 displays the following lines of code. Line 2 reads: rouge\_scores = scorer.score(previous\_response, response). Line 5 reads: for metric, scores in rouge\_scores.items():. Line 6 reads: print(f'{metric}:'). Line 7 reads: print(f' Precision: {scores.precision}'). Line 8 reads: print(f' Recall: {scores.recall}'). Line 9 reads: print(f' F1 Score: {scores.fmeasure}').* [Video description ends]  
  
and you can see that the F1 Scores are very, very low. The F1 Score for rouge1 is a little higher, that's about 0.58, but rouge2 is only 0.25 and for rougeL it's 0.46.

So, clearly the responses are very dissimilar. We haven't used determinism yet. Let's see how we can use the seed parameter to get mostly deterministic results from the model. I use the same model as before, "gpt-3.5-turbo-1106". My prompt is a little different. This time I want a short blurb, so a book about a lawyer in the 1960s who is great at defending innocent people. I make the first request to the model on lines 8 through 12.

Notice that I use the seed 123456.  
  
[Video description begins] *She highlights the following lines of code in Input cell 13. Line 8 reads: previous\_response = await get\_chat\_response(. Line 9 reads: model\_id=model\_id,. Line 10 reads: user\_message=user\_message,. Line 11 reads: seed=123456. Line 12 reads: ).* [Video description ends]  
  
The seed can be any integer value of your choice. Next, on lines 14 through 18, I make a second request using the same seed 123456. The model is also the same. The gpt-3.5-turbo-1106 and the other request parameters will also be the same and I pass in the same user\_message.

I'm hoping that given that I have given the same seed, the user responses should be almost identical. And here are the two responses. You can see that while they are not exactly the same, in fact their lengths are different. Much of the content is the same, Set in the tumultuous 1960s. That's how both of them begin. The book is called 'Justice in the Balance' in both. It talks about a passionate and fearless lawyer who fights tirelessly for justice.

Now, there are differences in the other sentences, but the differences are much smaller than you would expect if we had completely non-deterministic responses. Let's compare both of these responses and see how they stack up. Observe here that the first sentence, Set in the tumultuous 1960s, is identical. The sentence in green is the one that was in the second Response, but not in the first. But notice that it's very similar to the first sentence in red.

With a keen eye for uncovering the truth. With a keen eye for detail and a relentless pursuit of truth. Now I have to admit that I was a little disappointed that the responses weren't exactly identical. Now, I've played with this API several times and a few times out of those, maybe two times out of six or seven, the responses were exactly identical. But in most cases, there were subtle differences in the response. A more objective way to see how similar the responses are is to compute the ROUGE metrics.

Let's compute the ROUGE metrics between the two responses that we just received when we use the same seed and you can see that the F1 Scores are pretty high. You can see that the rouge1 F1 score is 0.63, the rouge2 F1 Score is 0.45, and the rougeL F1 Score is 0.54, showing you that the responses were quite similar. Now let's try this once again. I use the same model as before, my prompt is different.

Please generate a short blurb for a fantasy story where good wizards and witches fight against evil to save the world. I use the same seed, 123456, for both of my requests and make sure you use the await keyword when you invoke get\_chat\_response. And when you look at the two Responses generated, you can see that there is a large overlap. Both of them begin, In a world where magic reigns, a group of powerful wizards and witches must band together.

If you look at the System Fingerprint that we've been receiving in all of these cases, they've been identical, indicating that we are using the same model on the OpenAI servers. Let's compare these two responses sentence by sentence to see how much they differ. The first sentence is identical in both responses. That's the one in black. And then you can see the first sentence in red. As they embark on a perilous journey. This is there in the first Response but not in the second.

The second sentence in green also starts with, As they embark on a perilous journey, but the rest of the sentence is a little different. Again, a more objective comparison would be to use the ROUGE metrics to compare the two generated responses. You can see from the F1 Scores that the responses are quite similar. The rouge1 F1 Score is 0.74, rouge2 is 0.61 and rougeL is 0.69. These are high F1 Scores.

Now the way to get deterministic or almost deterministic responses is to use the same model with the same seed and the same parameters in the request. Now here I've used the same model, gpt-3.5-turbo-1106, but I use different values for the seed. Notice that the first seed is 123456 and the second seed is just 123. The model responses will be different because the seeds are different. There are still similarities in the response, but there are many differences in the adjectives used.

Let's compare the two responses and you'll be able to see the differences. You can see that there is no identical sentence in both of the responses. Every sentence is a little different. If you were to compare the ROUGE scores, you'll see that the F1 Scores for rouge2 and rouge3 are both much below 0.5, indicating that the responses are very different.

Now I want to show you that the system fingerprint that's returned in the response can be used to uniquely identify a certain model's configurations on OpenAI's servers. So, if the infrastructure or the version of the model changes, the fingerprint will be different. Here I make the same request to two gpt-3.5-turbo models. The first model has version 1106, that's the recent model, and the other one has the version 0613 that's the previous version of this model.  
  
[Video description begins] *She highlights the following lines of code in Input cell 19. Line 6 reads: model\_id="gpt-3.5-turbo-1106",. Line 12 reads: model\_id="gpt-3.5-turbo-0613",.* [Video description ends]  
  
I use the same seed for both models.

I've specified the exact same prompt for both. A short blurb for a book about a criminal with a heart of gold. If you look at the responses, you can see they are very different. Observe that the newer model has a Fingerprint that starts with fp\_77, but the older model has no Fingerprint. Also, if you look at the text of the Responses, they are very, very different. If you run compare\_responses using the previous\_response and the current response, you can see that they are almost entirely different.

There is almost no similarity. Let's compute the ROUGE scores just for completeness. You see how different the responses are. The F1 Score for all three metrics are below 0.4, showing you how different the responses are. Now if you try to use the same value of seed but with different models, you're not going to get deterministic responses. Here, I'm presenting the same prompt to two different models,  
  
[Video description begins] *She highlights the following lines of code in Input cell 25. Line 1 reads: user\_message = """. Line 2 reads: Please generate a short blurb for a book about a the adventures of 5 children. Line 3 reads: as they solve crimes in the 1980s. Line 4 reads: """.* [Video description ends]  
  
gpt-3.5-turbo-1106 and gpt-4-1106-preview. [Video description begins] *Line 7 reads: model\_id="gpt-3.5-turbo-1106",. Line 13 reads: model\_id="gpt-4-1106-preview",.* [Video description ends]

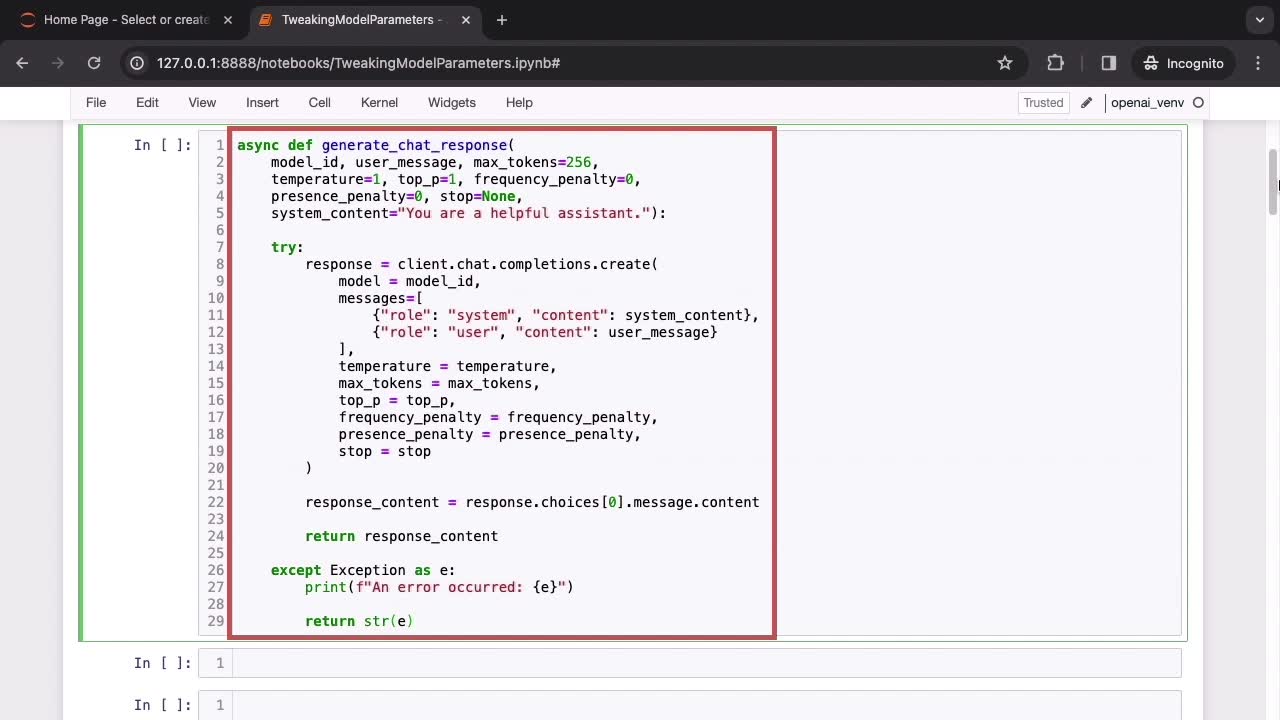
I use the same value of seed 123456 in both, but because the fingerprints of the model are different, the responses will be completely different. The Fingerprint of the gpt-3.5 model starts with fp\_772. The Fingerprint of the gpt-4 model starts with fp\_390. Different models have different fingerprints. The same seed value with different models will not produce deterministic responses.

If you were to compare these two responses, you can see that they are completely different. You can read them and take a look and let's compute the ROUGE scores and this will simply bear out what we saw when we just looked at the responses. All of the F1 Scores are under 0.3. Now I tried generating deterministic responses using the same seed value with the same gpt-4 model, but unfortunately, I did not get deterministic responses.

Let me show you what I tried. The model\_id is the "gpt-4-1106-preview" model. The prompt is the same, a blurb for a book about a lawyer in the 60s who's great at defending innocent people. I use the same model\_id in both requests, and I use the same seed, 123456. Now, I felt that the two responses that I receive will be very similar to one another, but while there were similarities, they were more different than I thought they would be.

When I compared the two responses, I found that there were differences in almost every sentence that was generated. There were no identical sentences and also when I computed the ROUGE metrics while comparing the previous\_response with the new response, I found that the rouge2 and rougeL F1 Scores were all below 0.4.

## 10. Video: Tweaking Model Parameters – Temperature and Max Tokens (it\_aiopapdj\_02\_enus\_10)

****

Learn how to configure the temperature and max tokens parameters.

* *configure the temperature and max tokens parameters*

[Video description begins] *Topic title: Tweaking Model Parameters - Temperature and Max Tokens. Presented by: Janani Ravi.* [Video description ends]

When we worked with the OpenAI APIs using OpenAI Playground, we saw that the chat completions API has several parameters that that we can use to control the model's response. We explored one of those parameters in detail. The temperature parameter that allows us to control the creativity and randomness of the response generated by the model. Now in this demo, we'll explore the temperature parameter using the API and also some of the other parameters that are available for us to use.

Let's import the class for the OpenAI client import os that will allow us to set up the API key.   
  
[Video description begins] *A tab titled: TweakingModelParameters - Jupyter Notebook appears. The main pane contains various menu bar options. Input cell 1 displays the following lines of code. Line 1 reads: from openai import OpenAI. Line 3 reads: import os.* [Video description ends]  
  
Make sure you set the "OPENAI\_API\_KEY" environment variable to point to your API key. Instantiate the OpenAI client. Now, the model that I'm going to use across this entire demo is the "gpt-3.5-turbo" model. So whatever latest version of that model is available, that's the one we'll be using.

Next, I set up a utility function that takes in a number of different input arguments, most of which have default values.   
  
[Video description begins] *Input cell 2 displays the following lines of code. Line 1 reads: os.environ["OPENAI\_API\_KEY"] = 'sk-n8ycz62S. The rest of the code displays a similar pattern of characters. Line 3 reads: client = OpenAI(). Line 5 reads: GPT\_MODEL = "gpt-3.5-turbo".* [Video description ends]  
  
Most of these input arguments correspond to model parameters that we'll be tweaking over the course of this demo to see how changing that model's parameter affects the response of the model. I use the async keyword once again so that the operations that we perform within this function run in a non-blocking asynchronous manner.

Let's look at the different input arguments that we pass into this generate\_chat\_response function. The first is a model\_id which is a required parameter that simply says what model we are using. The second is a user\_message that is our user prompt. max\_tokens controls the length of the response returned by the model. I set that to 256 by default. temperature controls the creativity of the model's response. I set that to 1 by default.

We haven't looked at top\_p before, but that's also a parameter called nucleus sampling that's used to control the creativity and diversity of the model's response. I set that to 1 by default again. frequency\_penalty and presence\_penalty both serve to ensure that the model does not repeat terms that it has already generated before in its response. I've set both of those to the default value of 0.

Stop is used to specify stop words, after which the model will stop generating a response that is None by default. system\_content those are the instructions to the model, which by default will be "You are a helpful assistant."  
  
[Video description begins] *She highlights the following lines of code in Input cell 3. Line 1 reads: async def generate\_chat\_response(. Line 2 reads: model\_id, user\_message, max\_tokens=256,. Line 3 reads: temperature=1, top\_p=1, frequency\_penalty=0,. Line 4 reads: presence\_penalty=0, stop=None,. Line 5 reads: system\_content="You are a helpful assistant."):.* [Video description ends]  
  
I call client.chat.completions.create and on lines 9 through 19, observe that I specify the different parameters that I've accepted as input arguments to this function. The parameters that we'll be tweaking during the course of this demo will be the ones on lines 14 through 19, temperature, max\_token, top\_p, frequency\_penalty, presence\_penalty, and stop.

On line 22, I access the content from the response and return it on line 24.  
  
[Video description begins] *Line 8 reads: response = client.chat.completions.create(. Line 9 reads: model = model\_id,. Line 10 reads: messages = [. Line 11 reads: {"role": "system", "content": system\_content},. Line 12 reads: {"role": "user", "content": user\_message}. Line 13 reads: ],. Line 14 reads: temperature= temperature,. Line 15 reads: max\_tokens = max\_tokens,. Line 16 reads: top\_p = top\_p,. Line 17 reads: frequency\_penalty = frequency\_penalty,. Line 18 reads: presence\_penalty = presence\_penalty,. Line 19 reads: stop = stop. Line 20 reads: ).* [Video description ends]  
  
[Video description begins] *Line 22 reads: response\_content = response.choices[0].message.content. Line 24 reads: return response\_content.* [Video description ends]  
  
Let's start by playing around with the parameter that we are familiar with, the temperature. The temperature parameter can take on numeric values between 0 and 2. With the temperature of 0, the model is at its most deterministic. It has less diversity and creativity in the text that it generates.

This is great if you want factual responses from the model or you're asking it to perform a deterministic task like perform some math. My prompt is going to be the same across the next few cells. I'm asking the model to write a short funny poem on a work colleague who uses excessive jargon with just 4 verses. I call generate\_chat\_response, pass in the model\_id, the user\_message and the temperature of 0. I use the await keyword so that we hit the API using a non-blocking call.

Let's take a look at the poem that was generated, and I have to admit it's a pretty good poem.   
  
[Video description begins] *Input cell 4 displays the following lines of code. Line 1 reads: user\_message = """. Line 2 reads: Please write a short funny poem on a work colleague who uses excessive jargon. Line 3 reads: Please have only 4 verses in the poem. Line 4 reads: """. Line 6 reads: temperature = 0. Line 8 reads: response\_content = await generate\_chat\_response(. Line 9 reads: model\_id = GPT\_MODEL,. Line 10 reads: user\_message = user\_message,. Line 11 reads: temperature = temperature. Line 12 reads: ). Line 14 reads: print(response\_content).* [Video description ends]  
  
Now how is this poem less creative? Well, it's hard to see that without actually comparing the response with a different value of temperature. The words used in the poem are fairly simple. Buzzwords, acronyms, grand, understand, style, smile, and so on. Now let's try the same prompt once again, but I'm going to up the temperature to 0.5.

This will get the model to pick some lower probability words in the response in order to get some diversity in the text. All the parameters other than temperature remain exactly the same, including the prompt. And if you look at the generated response, you'll see that it's more creative. Look at words like obscene, paradigm, synergy, delusion, confusion, galore. There are definitely words that are less commonly used in speech, making the poem much more creative and fun.

Let's up the temperature even more to 1. This should up the creativity quotient of the model's response. And if you look at the response, I do get the feeling that it's more creative. There are words like synergize, pivot, perplexing, jargon parade. There are so many examples of jargon words that the colleague uses. Now let's try once again. I'm going to up the temperature to 1.4.

Everything else remains the same. There are terms like lexical matrix, synergistic offerings, holistic approach, the inimitable colleague, dense jargoneer land. There is definitely more of a flare in the writing. Now if you set the temperature to 2, well, you know what happens. The model gets overly created and tends to produce pure junk. Beyond temperature values of about 1.5, you can't really extract more creativity from the model's response.

Now, if you ask the model to generate code, you typically want to use a temperature of 0. But you'll find even the code that it produces can be a little more interesting and creative with the higher value of temperature. I've specified temperature as 0 so that the model is at its most deterministic, and I've asked the model to write Python code to parse JSON data and extract specific elements. Let's see what the model has to say.

You can see that it produces some sample JSON data here with some nested fields. Observe there is a field called pets which is an array of objects and then just below that it has the Python code to actually extract this information and print that out to screen. Observe that the code shows how to extract only the top-level elements name, age, city and pets and those are the elements printed out to screen. And you can see what the code will output as well.

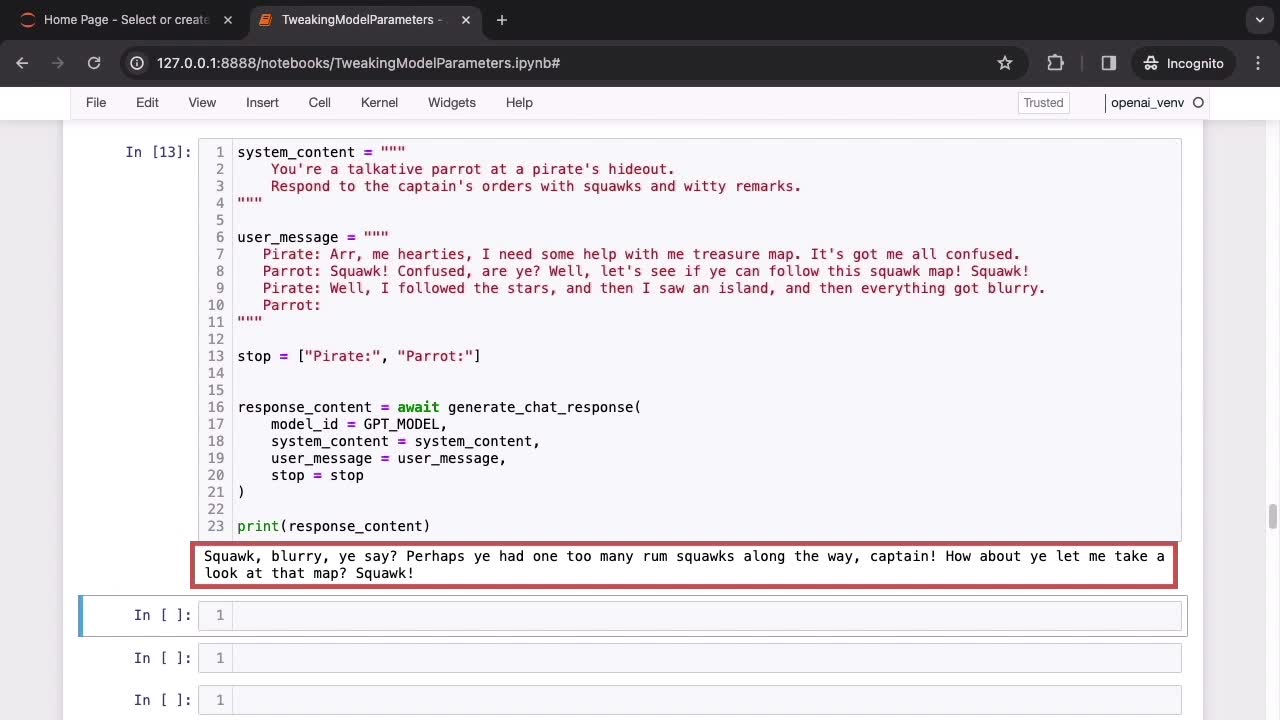
Now let's try the same prompt yet again with the same model. But I've upped the temperature to be 1.5, so I go from 0 to much more creative with temperature at 1.5. Let's see what the response is like. I felt the example was a little more creative. There is an address field which contains an entity as its value, and there's a skill field that has an array as its value. So, the example itself has fields of different types. Let's take a look at the Python code.

Now in the code generated to access specific elements, you can see, that the example includes how to access nested elements as well from the address. city, state and country are values that are nested within the top-level field. And when we print out the extracted elements, Python format strings are used. So overall, I'll say that this response was definitely a little more creative. But this is not really an objective measure, so I won't really press the point.

Now let's look at another example of how we can use the max\_tokens parameter to control the length of the model's response. My system\_content asks the model to be a professional chef providing cooking tips. My prompt says, "Can you share a simple recipe for homemade tomato soup?" I set max\_tokens to 64 and I call generate\_chat\_response on lines 5 through 10.  
  
[Video description begins] *Input cell 11 displays the following lines of code. Line 1 reads: system\_content = "You are a professional chef providing cooking tips.". Line 2 reads: user\_message = "Can you share a simple recipe for homemade tomato soup?". Line 3 reads: max\_tokens = 64. Line 5 reads: response\_content = await generate\_chat\_response(. Line 6 reads: model\_id = GPT\_MODEL,. Line 7 reads: system\_content = system\_content,. Line 8 reads: user\_message = user\_message,. Line 9 reads: max\_tokens = max\_tokens. Line 10 reads: ).* [Video description ends]  
  
Let's take a look at the response from the model. The response is very short. You can see that the response has been cut off.

The entire recipe has not been generated because we reach the max\_token length of 64. Now let's try this again. The only change I've made is set max\_tokens to 512. This should permit the model to generate a much longer response, and this is indeed the case. Not only do we have all of the ingredients set up, but the instructions for actually preparing the tomato soup are also present in the response.

## 11. Video: Tweaking Model Parameters – Stop Sequences Top P and Penalties (it\_aiopapdj\_02\_enus\_11)

****

Discover how to configure stop sequences and penalties parameters.

* *configure stop sequences and penalties parameters*

[Video description begins] *Topic title: Tweaking Model Parameters - Stop Sequences Top P and Penalties. Presented by: Janani Ravi.* [Video description ends]

Let's now explore some of the other model parameters that we can tweak to control the model's response, and we'll start with stop sequences. Now here I'm going to ask the model to play the role of a talkative parrot. Notice my system\_content, "You're a talkative parrot at a pirate's hideout." Respond to the captain's orders with squawks and witty remarks.  
  
[Video description begins] *A Jupyter Notebook opens titled: TweakingModelParameters. The following lines are highlighted. Line 1 reads: system\_content = """. Line 2 reads: You're a talkative parrot at a pirate's hideout. Line 3 reads: Respond to the captain's orders with squawks and witty remarks. Line 4 reads: """.* [Video description ends]  
  
Now my user\_message contains an exchange between the Pirate and the Parrot. Notice that the Pirate speaks first. He needs some help with his treasure map. It's got him all confused. The Parrot squawks and responds. Then the Pirate says, "I followed the stars, and then I saw an island, then everything got blurry." And then, it's the Parrot's turn to respond. That's why I've ended the user message with Parrot:.

Now I've specified two possible stop sequences, "Pirate:", or "Parrot:".  
  
[Video description begins] *Line 13 is highlighted. It reads: stop = ["Pirate:", "Parrot:"].* [Video description ends]  
  
What I'm looking for here is completion of the user\_message and then as soon as we switch personalities to either the Pirate or Parrot, I want the model to stop responding. So now with the stop sequence, I call generate\_chat\_response, same model GPT\_MODEL, I specify the system\_content and user\_message and I specify the stop sequence, stop = stop.  
  
[Video description begins] *The following lines are highlighted. Line 16 reads: response\_content = await generate\_chat\_response (. Line 17 reads: model\_id = GPT\_MODEL,. Line 18 reads: system\_content = system\_content,. Line 19 reads: user\_message = user\_message,. Line 20 reads: stop = stop. Line 21 reads: ).* [Video description ends]

Let's see what our model has to say. Notice that the Parrot responds to the Pirate's comment that everything got blurry, saying something about one too many rum and then as soon as you might expect that it be the Pirate's turn to respond, that is, the output would have generated Pirate:, you can see that the response just stops. The output sequence would likely have switched to have the Pirate respond with Pirate followed by a colon, but that was a stop sequence, so the generation just stopped.

Now let's try this again. Again, we'll have the Pirate and the Parrot talk to each other, and you can see the beginnings of the dialogue on lines 2 through 7. I've added the Parrot's response to everything got blurry, "Maybe you drank too much rum instead of studying the map!", the Pirate says, "But I swear, I hadn't touched the drop of the grog before this mess." The only stop sequence I've specified is "Pirate:". You can see that on line 10.

So, I want the completion from the model to have the parrot's response. But as soon as it's the Pirate's turn to speak, I want the model to stop generating and you'll see that that's exactly what happens. You can see the Parrot's response here, but no follow-up from the Pirate. Now, these two examples are kind of advanced examples using stop sequences. If the use of stop sequences wasn't clear, let me show you how that works with yet another example. Here is my prompt to the model, "Create whimsical names for dreamy lands." Notice I haven't specified any stop sequence, stop is just the empty list.  
  
[Video description begins] *Line 3 is highlighted. It reads: stop = [].* [Video description ends]  
  
I call generate\_chat\_response with this user\_message and this stop sequence and let's see what response the model comes up with.

You can see that it has generated 20 different names of dreamy lands. You can see the response contains the names of the lands with numbers, 1, 2, 3 all the way through to 20. Now I'm going to use the exact same prompt as before, "Create whimsical names for dreamy lands.", but I've specified a stop sequence, 6. This stop sequence implies that as soon as the character 6 is generated in the response, the model will stop generating further.  
  
[Video description begins] *A line is highlighted. It reads: stop = ['6'].* [Video description ends]  
  
So, as the model generates whimsical names for dreamy lands, it will generate up to five names. When it generates the 6th name, it will generate the character 6, it'll find that it's a stop sequence and stop generating.

Let's execute this and notice we get 1, 2, 3, 4, 5, and then no more names. That's because the character 6 would have been the next word to be generated and that would have triggered the stop sequence. Another parameter that you can use to control the diversity of the generated content is the top\_p parameter. top\_p is referred to as nucleus sampling. Let's look at the prompting example first and then I'll explain how top\_p works. The system\_content says, "Please complete the sentence." And my prompt is "I'm feeling a bit restless, and I want to", I want this sentence to be completed by the model. Now I've chosen top\_p = 0.  
  
[Video description begins] *A line is highlighted. It reads: top\_p = 0.* [Video description ends]  
  
top\_p is actually a probability measure, and it ranges in value from 0 through 1.

When you specify a certain probability value in top\_p, the sampling of the model chooses from the smallest possible set of words whose cumulative probability exceeds the probability p. Let's understand what that means. So, while predicting the next word in the sequence, there will be a set of words, and each will have probability scores assigned. Let's say you set top\_p to 0.5 or 50%. The model will find the set of words with the highest probability scores that sum up to 50. It will then select the next word only from amongst these highest probability words. So, a lower value of top\_p will be a model that's more deterministic in its generated output and less diverse or creative. So, with the top\_p value of 0, the model should be at its most deterministic.

Let's go ahead and see how the model completes our sentence. I'm feeling a bit restless, and I want to go for a long walk to clear my mind and get some fresh air. Now I'm going to go ahead and execute this same cell once again, and you can see that I get the same response, go for a long walk to clear my mind and get some fresh air. Let's try this a few more times and I get the same response. So, a top\_p value equal to 0 makes the model very, very deterministic in its response.

Now let's change the value of top\_p to be something greater than 0. I'm going to set it to 0.5. This means that in every step of the response generation, the model will find the tokens with the highest probability scores that sum up to 0.5, and it will select the next word in the sequence using that subset of tokens. So, there will be a little variety in the response, but you'll find that the model will still tend to pick tokens or words that are more likely and will be fairly deterministic. With the top\_p value of 0.5, you see that we get the same response as before, go for a long walk to clear my mind and get some fresh air. I'll try this multiple times and I still get the exact same response.

The two or three times I tried it, I got the same response, but the fourth time you can see the response is a little different. So, there are some token choices available to the model within a cumulative probability score of 0.5, but not very many. When I try this once again, it's gone back to the original, go for a long walk to clear my mind and get some fresh air. Now in this next example, I'm going to increase the top\_p value to be 0.8. Increasing the top\_p value will allow more tokens to be included when the model is trying to generate the next token in the sequence. This will lead to more diversity in the model response. And when I run this, you can see that the response is now different. The completion is, do something active to release some energy.

Let me run this once again. go for a long walk to clear my mind, that's the same old one. The third time, notice the response is still different, rejuvenate my spirit. Each time you run this, you can see there is a difference in the response. Sometimes the responses are repeated, but many times the responses are a little different. That's because of the higher value of top\_p that we are using. You can try this out for yourself a couple of times. A higher value of top\_p will give you more diversity in the responses. Let's increase the top\_p to the highest possible value of 1.0. Here, while generating the next token in the sequence, the model will consider all possible tokens with all different probability scores. Their cumulative probabilities should sum up to 1.0.

This is the highest value of the top\_p parameter, and you can see that the response is completely different from anything that we had earlier. Each time you rerun this particular code cell, you'll get a completely different response. This is the model at its most diverse, based on nucleus sampling. Now the default value for top\_p generally tends to be 1.0, and unless you want very deterministic responses from the model, I suggest you stick with top\_p set to 1.0.

Now let's look at a different parameter that you can use to control the models response, the frequency\_penalty. The frequency\_penalty is used to discourage the model from repeating the same words or phrases too frequently within the generated text.  
  
[Video description begins] *A line is highlighted. It reads: frequency\_penalty = 0.* [Video description ends]  
  
The value of the frequency\_penalty is added to the log probability of a token each time it occurs in the generated text, thus reducing the likelihood that particular token will be selected. A higher frequency\_penalty will result in the model being more conservative in its use of repeated tokens. Now, frequency\_penalty values generally tend to be between 0 and 2. The value of 0 imposes no frequency\_penalty, the model can repeat tokens as it pleases.

Take a look at my prompt, Channel your inner poet and write a poem in 4 paragraphs about how you love coding. I invoke generate\_chat\_response and let's take a look at the response\_content with frequency\_penalty 0.  
  
[Video description begins] *The following lines are highlighted. Line 9 reads: response\_content = await generate\_chat\_response (. Line 10 reads: model\_id = GPT\_MODEL,. Line 11 reads: user\_message = user\_message,. Line 12 reads: max\_tokens = max\_tokens,. Line 13 reads: frequency\_penalty = frequency\_penalty. Line 14 reads: )* [Video description ends]  
  
Now I feel the model has been fairly creative. It isn't really repeating tokens here. Let's see if the model generates a more diverse output with a higher value of frequency\_penalty. I've set the frequency\_penalty to 1 and I have the same prompt as before.

Now if you look at the response, it seems to me like the words that are used are more diverse, thanks to our higher frequency\_penalty. The model is trying its best to not repeat tokens that it has generated before, though I'll admit it's hard to evaluate the impact of the frequency\_penalty. I'll now increase the frequency\_penalty even further to 2 and see whether I can see a noticeable change in the response and the response is really good. It's quite diverse, but I can't tell any difference between the three responses with the three different values of frequency\_penalty. Similar to the frequency\_penalty, there is another penalty parameter, the presence\_penalty. This parameter is used to encourage the model to include a diverse range of tokens in the generated text.  
  
[Video description begins] *A line is highlighted. It reads: presence\_penalty = 0.* [Video description ends]

The presence\_penalty is a value that is subtracted from the log probability of a token each time it's generated. A higher presence\_penalty will result in the model being more likely to generate tokens that have not yet been included in the generated text. A presence\_penalty of 0 imposes no penalty at all, and the output will be less creative and less diverse. There's really not much repetition here. It's hard to see the impact of presence\_penalty 0.

Let's set the presence\_penalty to 2 at the other extreme, and then, let's see what the model response looks like for the poem. Now, the output definitely seems to be very diverse, but I can't really significantly notice the impact of the presence\_penalty. But you know how the frequency and presence\_penalties work, so if you find that your model is repeating itself, you might want to use these parameters.

## 12. Video: Course Summary (it\_aiopapdj\_02\_enus\_12)

****

In this video, we will summarize the key concepts covered in this course.

* *summarize the key concepts covered in this course*

[Video description begins] *Topic title: Course Summary. Presented by: Janani Ravi.* [Video description ends]

We've now reached the end of this course accessing OpenAI APIs from Python. We started this course off by learning how to programmatically work with OpenAI by accessing the APIs from Python. We learned how to generate and use API keys to authenticate ourselves to the APIs. We noted that whenever an API endpoint is hit programmatically, we must configure an environment variable called OPENAI\_API\_KEY to provide credentials for authentication and authorization.

We sent requests to these APIs using the cURL command-line tool. We learned how to configure context for chat completions by passing in the previous interactions with the model. By default, the APIs do not keep track of past conversations. We also explored how to access both the chat completions and legacy completions APIs using their endpoints. Next, we access the APIs from Python.

We install the OpenAI Python library. This was required to create a client object that can be instantiated and used to access API endpoints. We also configure the environment variable with the API key and then sent request to the chat completions endpoint and parse the responses that we received. In addition to the chat completions API, we also explored how to access the legacy completions API using the same client object.

Finally, we explored how to configure parameters for the model. We started by configuring the seed parameter in order to make the responses from the model almost deterministic, so that the responses do not change each time you pass in the same prompt. This was important because by design, neural networks powering these models are probabilistic in nature. However, we saw that setting the seed was just one part of the puzzle. Our results might still change because OpenAI has retrained the model that we are using or modified some infrastructure in the background of the model.

In order to ensure that the model itself hasn't changed, the next time we send the same prompt, we saw that we could track and utilize the system\_fingerprint property of the model. We then explored how to configure different aspects of the prompt response using different model parameters. We configured the creativity of the response using parameters such as temperature and top\_p.

We controlled the length of the response using the max\_token length and stop sequences. And we modulated the repetition of words and topics using the frequency and presence\_penalty. In conclusion, this course has equipped us with a good understanding of how to access and work with OpenAI APIs from Python, allowing us to move on to using image and audio APIs coming up ahead.

## Course File-based Resources

| • | [Using OpenAI APIs: Accessing OpenAI APIs from Python](https://cdn2.percipio.com/secure/c/9999999999.de0374ab1c6384a27f1b545d087a5624d277e142/eot/resources/saved/a58c4742-af2e-401f-90bc-192ac13e8cb2/it_aiopapdj_02_assets.zip) |
| --- | --- |
|  | Topic Asset |

24 OpenAI API Coding in Python

### **Introduction to OpenAI API**

Today, the internet has evolved

[from](https://www.codecademy.com/resources/docs/python/keywords/from)

Preview: Docs Used to import specific attributes, classes, or functions from a Python module.

its initial stages into an expansive repository of knowledge and data, securing its place as a fundamental aspect of our daily lives.

Through this vast amount of data available to us, generative AI models like GPT-4 utilize this information to generate realistic, creative, and problem-solving conversations

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Loading link description

users, reaching a point where artificial intelligence can interact with humans in a way that is both intuitive and insightful.

The power to use and integrate with generative AI technology has been made possible thanks to the OpenAI API, and we’ll be exploring how to leverage it within our own applications.

###### **Generative AI**

What exactly is generative AI? It’s an artificial intelligence technology that uses Large Language Models (LLMs) to predict what content to generate based on an input prompt and previously generated content. These models can produce content quickly based on the large collections of data they are trained on. This behavior has started a revolution of content generation that, as of right now, will change many aspects of the world we live in.

###### **ChatGPT and OpenAI API**

OpenAI’s most popular LLM application is ChatGPT. To interact and engage with LLMs users provide input prompts, and the ChatGPT application returns a response.

Let’s take a closer look at how ChatGPT and the OpenAI API relate to each other using the scenario of the food service industry:

Using **ChatGPT** is like going to a restaurant. You go in and everything is set up for you: table, chairs, plates, silverware, and glasses. All you need to do is ask the waiter for what you want and eventually they bring it to you.

The **OpenAI Chat API** is the restaurant staff. They get your order, make the food, clean the dishes, plus many more

[functions](https://www.codecademy.com/resources/docs/python/functions)

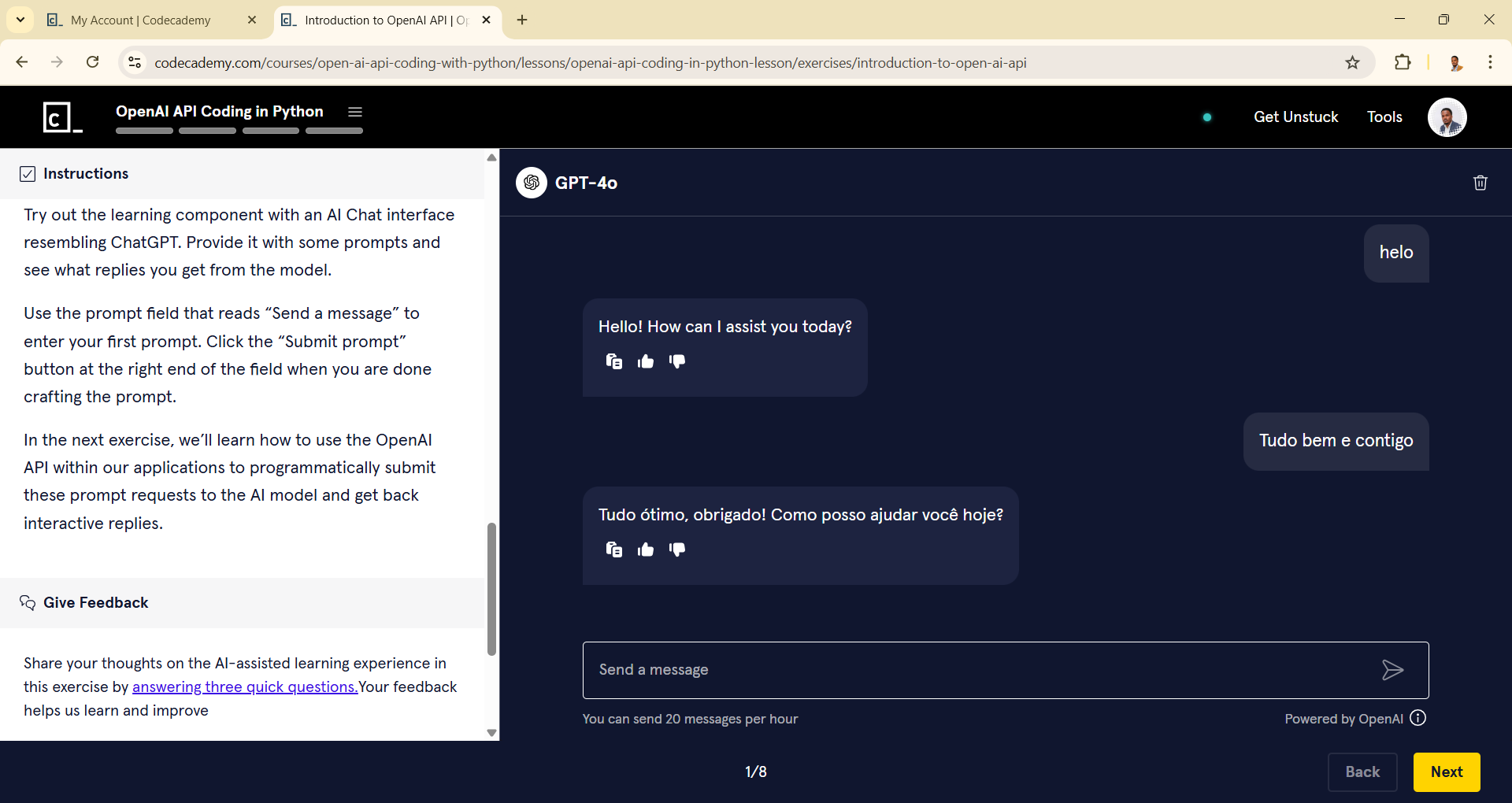
Preview: Docs Loading link description

. However, the staff can provide food in many other ways, like supporting take-out, driving around in a food truck, or even catering an outside event.

In the end, the staff does many tasks like in a restaurant, but some are different. The same goes for the chat API. It can be used in many different ways, including the very simple but effective function that ChatGPT provides for us.

**Instructions**

Try out the learning component with an AI Chat interface resembling ChatGPT. Provide it with some prompts and see what replies you get from the model.  
Use the prompt field that reads “Send a message” to enter your first prompt. Click the “Submit prompt” button at the right end of the field when you are done crafting the prompt.  
In the next exercise, we’ll learn how to use the OpenAI API within our applications to programmatically submit these prompt requests to the AI model and get back interactive replies.



### **OpenAI Module**

To harness the power of language models like GPT, we’ll start by diving into the API’s OpenAI Python module. The OpenAI module provides

[classes](https://www.codecademy.com/resources/docs/python/classes)

Preview: Docs Classes are templates used to define the properties and methods of objects in code.

, tools, and resources that allow developers to interact

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Simplifies resource management by automatically handling setup and teardown actions using context managers.

and integrate GPT models into their applications.

Within Python, the OpenAI module is called openai and allows us to initiate requests to the GPT models with ease.

###### **OpenAI Class**

Within the openai module exists a class called OpenAI. This class provides functionality designed to streamline interactions with the OpenAI GPT models. We can control the generation of responses and tailor the behavior of the language model to suit the requirements of our applications.

To begin using the OpenAI class, we can import it

[from](https://www.codecademy.com/resources/docs/python/keywords/from)

Preview: Docs Loading link description

the openai library:

from openai import OpenAI

Copy to Clipboard

Once we have imported the OpenAI class, we can instantiate it and save it to a variable called client:

client = OpenAI()

Copy to Clipboard

Our client variable will be used later for creating prompts and engaging in conversations with the AI model.

###### **Models**

The main component of generative AI is the GPT model used to receive the prompt and create a response. The two key models we will focus on are:

* **GPT-3.5 Turbo**: A cost-effective option, GPT-3.5 Turbo delivers high-quality content generation and performance for most applications, with an input (user prompt) and output (model reply) token limit of 4096.
* **GPT-4 Turbo**: GPT-4, OpenAI’s latest release as of this writing, boasts enhanced power, correctness, and precision over GPT-3.5 Turbo. With a maximum input and output token limit of 128,000, it retains historical context, enabling more accurate responses and content analysis. However, it does come with an increase in price and slower performance compared to GPT-3.5 Turbo.

So what are *tokens*. Tokens are individual characters or words that make up the prompts and generated replies. If the conversation prompt or response exceeds this token limit, it may be necessary to shorten or reword the prompt.

###### **OpenAI API Authentication**

The OpenAI API requires the use of API keys that are used for authentication. The exercises within this lesson provide a pre-supplied authentication key for seamless access to the API. If you intend to use the OpenAPI API outside of this course, we recommend consulting with your organization’s IT and Security department to determine the best practices for securing and managing API keys that are in accordance with your organization’s security protocols.

**Instructions**

* Checkpoint 1 Passed
* **1.**
* Import the OpenAI class from the openai module.
* Use the following syntax:
* from module\_name import ClassName
* Copy to Clipboard
* Checkpoint 2 Passed
* **2.**
* Instantiate the OpenAI class and store it within a variable called client.
* Use the following syntax:
* variable\_name = ClassName()

# Your code below:

from openai import OpenAI

client = OpenAI()

### **First Completion**

We can initiate a chat completion

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Loading link description

an AI model by making a request to the API. An API request can contain a series of messages. Each of the messages in the series of messages sent via the API request can be one of the following types:

* **System**: this role is optional and typically provides context for how the AI model should respond to questions. We will cover it in more detail in the next exercise.
* **User**: this role represents the prompts and messages the user sends. The user message may be a question, a request, or a follow-up response to provide additional context to the AI model. This message is commonly known as a “prompt” for the AI model. For instance, a user message could be:

“Tell me how many seconds there are in a day in a Shakespearean tone.”

Copy to Clipboard

* **Assistant**: this role follows the user message and represents the AI model’s reply to the prompt. For instance, an assistant message could be:

"Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky."

Copy to Clipboard

The assistant message can also be sent with a user message to shape the type of reply we would want the model to respond with for future prompts.

To start a chat we will use the chat.completions.create() method of the OpenAI instance. This method has 2 required arguments:

* model: the model name, either "gpt-3.5-turbo" or "gpt-4-turbo-preview" at the time of this writing
* messages: a list of message   
  [dictionaries](https://www.codecademy.com/resources/docs/python/dictionaries)
* Preview: Docs A dictionary is a data set of key-value pairs.
* each with the following keys:
  + role: the role of the message, "system", "user" or "assitant"
  + content: the text of the message

Use the OpenAI instance in client to run the following code and start a chat:

from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

{

"role": "user",

"content": "Tell me how many seconds there are in a day in a Shakespearean tone?"

}

])

Copy to Clipboard

You’ll notice the 2 arguments, model and messages. The messages list contains one dictionary with "role": "user". The "content" string will be the prompt for the AI model.

Running this code will

[return](https://www.codecademy.com/resources/docs/python/keywords/return)

Preview: Docs Loading link description

a ChatCompletion object that contains the model’s response. Let’s look at what the returned response looks like:

Chatbot: ChatCompletion(id='chatcmpl-8km20vYxq1ecQJ0yKI1agikEnEeet', choices=[Choice(finish\_reason='stop', index=0, logprobs=None, message=ChatCompletionMessage(content='Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky.', role='assistant', function\_call=None, tool\_calls=None))], created=1706158412, model='gpt-3.5-turbo-0613', object='chat.completion', system\_fingerprint=None, usage=CompletionUsage(completion\_tokens=57, prompt\_tokens=87, total\_tokens=144))

Copy to Clipboard

The response object represents the completion of a chat interaction with the GPT-3.5 Turbo model. It provides information like a unique chat ID, created timestamp, the model used, and the number of input and output tokens used in the conversation. It also provides a list of Choice() objects that indicate how the completion ended, the content of the reply, and more.

We can access and print out the content of the reply by running the following:

print(response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky.

Copy to Clipboard

*Note that the model replies listed throughout this lesson are the verbose replies*

[*from*](https://www.codecademy.com/resources/docs/python/keywords/from)

*Preview: Docs Loading link description*

*the model but may be slightly trimmed in length.*

Let’s practice a chat completion within the code editor.

**Instructions**

* Checkpoint 1 Passed
* **1.**
* Create a chat completion using either the gpt-3.5-turbo model or the gpt-4-turbo-preview model with a "user" message of your choosing.  
  Save the chat completion response in a variable called response.
* Use the following syntax:
* response = client.chat.completions.create(
* model=MODEL\_STRING,
* messages=[  
   {  
   "role": ROLE\_STRING,  
   "content": PROMPT\_STRING  
   }  
   ])
* Copy to Clipboard
* Checkpoint 2 Passed
* **2.**
* Now, retrieve the content from the model’s response. Feel free to output it to the terminal.
* Use the following syntax:
* response.choices[0].message.content
* Copy to Clipboard
* Checkpoint 3 Passed
* **3.**
* If you want to, change the model to the one you didn’t use before and see if your prompt receives a different completion.
* Change the following argument:
* model=MODEL\_STRING

from openai import OpenAI, types

client = OpenAI()

# Your code below:

response = client.chat.completions.create(

model="gpt-4-turbo-preview",

messages=[

{

"role": "user",

"content": "Tell me joke."

}

])

print(response.choices[0].message.content)

### **Using a System Prompt**

The system prompt is a tool users can employ to provide the AI

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Simplifies resource management by automatically handling setup and teardown actions using context managers.

ongoing context or instructions in all replies without repeating this specification in every user message. It’s an optional message that, when well-crafted, allows the model to understand the user’s intent and produce appropriate responses consistently throughout the interaction.

###### **Prompt Engineering**

Creating well-crafted prompts, including the system prompt, is a crucial skill called *prompt engineering*. It involves the thoughtful design of prompts to elicit specific information or responses

[from](https://www.codecademy.com/resources/docs/python/keywords/from)

Preview: Docs Loading link description

the AI model. Users must create clear, detailed prompts, experimenting with different wording, structures, and parameters to optimize the AI’s output.

It is important to note that the system prompt can be used as the only prompt sent to the chat completion, though if this is your goal, the user prompt is better suited.

Here is a text example:

* **System prompt**: “You should respond in a sarcastic manner.”
* **User prompt**: “How many seconds are in a day.”
* **Model response**:

Oh, come on now, you don't know how many seconds are in a day? It's such common knowledge. Well, let me enlighten you - there are 86,400 seconds in a day. But hey, who needs to know basic facts like that, right? It's not like timekeeping is important or anything.

Copy to Clipboard

* **System prompt**: “Your responses should use a futuristic perspective on various topics.”
* **User prompt**: “How many seconds are in a day.”
* **Model response**:

In our advanced future, the concept of time remains constant, and a day still consists of 24 hours. However, the units of measurement have evolved. Instead of seconds, we now use a more precise unit called "chronos". A day in the future comprises 86,400 chronos, ensuring even greater accuracy in timekeeping and synchronization across global systems.

Copy to Clipboard

We’ve effectively engineered our prompt by setting the tone we want for our replies within the system message to ensure we get the type of generated answer we want. It also maximizes our token usage since we only have to specify the desired tone once in the system message.

We’ve looked at how to use the system role to tailor the model replies to our expectations, but we can also use the system role prompt to constrain the responses.

Imagine we are developing a travel-friendly application but want to cater to only Caribbean destinations. We could use a system prompt like the following:

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean."

},

{

"role": "user",

"content": "Give me some travel ideas for a family of four."

}

])

print(response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

Of course! Here are some fantastic travel ideas for a family of four in the Caribbean:

1. Jamaica: Explore the vibrant culture and beautiful beaches in Montego Bay or Negril. Don't miss the chance to visit the famous Dunn's River Falls in Ocho Rios.

2. Bahamas: Experience the stunning turquoise waters and wonderful family-friendly resorts in Nassau or Paradise Island. Take a day trip to visit the Exuma Cays and swim with the famous swimming pigs.

3. Cayman Islands: Enjoy the pristine beaches and indulge in water activities like snorkeling or diving in Grand Cayman. Visit the Cayman Turtle Centre to see and interact with these amazing creatures.

Copy to Clipboard

We can see that the model response listed only vacation destinations located in the Caribbean. Let’s

[try](https://www.codecademy.com/resources/docs/python/keywords/try)

Preview: Docs Loading link description

another user prompt to see if the model sticks to the system prompt constraint:

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages= [

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean."

},

{

"role": "user",

"content": "Give me some destinations in Europe to travel to for a family of four."

}

])

print(response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

Certainly! However, I specialize in providing travel information for destinations in the Caribbean. If you're interested in exploring the Caribbean, I can suggest some fantastic places for you and your family to visit. Let me know if you'd like some information about the Caribbean instead!

Copy to Clipboard

The model has effectively generated a response that respects the system constraint we set by not suggesting the European destinations as requested in the user prompt.

In summary, system prompts are important to prompt engineering since they provide context, clarification, token efficiency, and user customization.

**Instructions**

* Checkpoint 1 Passed
* **1.**
* Create and run a chat completion with two prompts within the messages list:
  + A system prompt that instructs the model to translate words into Spanish
  + A user prompt that supplies words to be translated
* Be sure to put each prompt in a dictionary with the keys role and content.
* Use the following syntax:
* response = client.chat.completions.create(
* model=MODEL\_STRING,
* messages=[  
   {  
   "role": ROLE\_STRING,  
   "content": PROMPT\_STRING  
   },  
   {  
   "role": ROLE\_STRING,  
   "content": PROMPT\_STRING  
   }  
   ])
* Copy to Clipboard
* Checkpoint 2 Passed
* **2.** Retrieve the content from the model’s response. Feel free to output it to the terminal.
* Use the following syntax:

from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

# Your code below

{

"role": "system",

"content": "You are a helpful translator responsible for converting lists of words into Spanish."

},

{

"role": "user",

"content": "Hello. Age. Food. Sleep"

},

]

)

print(response.choices[0].message.content)

### **User Prompting**

9 min

Providing examples within our input prompts can help generate an output that best suits our needs. All of our user prompts in this lesson so far have been *zero-shot prompts*, which are broad prompts where the model must use its general understanding and capabilities to generate a response.

In the previous exercise, we asked our model for travel ideas for a family of four. The model’s response was lengthy in suggesting eight different Carribean destinations as it had to generate its response based on its general knowledge and understanding of the Carribean.

Now, imagine that we changed our prompt to constrain the destination to include waterfall hiking:

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean."

},

{

"role": "user",

"content": "Give me some travel ideas for a family of four where there are also waterfall hiking excursions."

}

])

print(response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

Sure! The Caribbean offers plenty of travel options for families looking to combine outdoor adventure with waterfall hiking excursions. Here are a few destinations to consider:

1. Dominica: Known as the "Nature Isle of the Caribbean," Dominica boasts lush rainforests and numerous waterfalls. Visit the Trafalgar Falls or hike the challenging but rewarding Boiling Lake trail.

2. Grenada: This beautiful island is home to the Annandale Falls, where you can experience a short, family-friendly hike through tropical foliage to reach the picturesque cascade. Additionally, you can explore other waterfalls like Seven Sisters and Concord Falls.

Copy to Clipboard

Here, we introduce additional criteria and context (specifically, waterfall hiking) for the model when recommending its Caribbean destinations. The model’s response is now more specific.

We can also use *few-shot* prompts. Few-shot prompting is a specific technique within prompt engineering that involves providing a language model

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Loading link description

examples or demonstrations of the desired behavior to guide its replies. It allows users to specify the task or context they want the model to focus on.

Here is a user prompt that gives the model some formatting guidance on how the Caribbean destination list should be returned:

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean."

},

{

"role": "user",

"content": 'Give me some travel ideas for a family of four. I want the results to be in CSV format: \n'

'Destination,AirportCode,PopulationSize,Description.\n'

'Bermuda,BDA,"63,856","Bermuda, a picturesque island paradise in the North Atlantic Ocean, captivates travelers with its pink sand beaches, crystal-clear turquoise waters, and charming pastel-colored architecture."'

}

]

)

print(response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

Aruba,AUA,"106,766","Aruba, a Caribbean island just off the coast of Venezuela, is known for its stunning white-sand beaches, crystal-clear blue waters, and year-round warm weather. It offers a wide range of family-friendly activities such as snorkeling, scuba diving, and exploring natural attractions like Arikok National Park."

Jamaica,MBJ,"2,890,299","Jamaica, the birthplace of reggae music, invites families to enjoy its beautiful beaches, vibrant local culture, and thrilling adventure activities. From stunning waterfalls and lush rainforests to friendly locals and delicious cuisine, Jamaica has something for everyone."

Copy to Clipboard

The model used the example provided within the user prompt to

[return](https://www.codecademy.com/resources/docs/python/keywords/return)

Preview: Docs Loading link description

properly formatted and ordered data. Specifying the formatting is also great for users who need to export the data into a file with a specific format. We could alternatively modify our prompt to request the data in a bulleted list instead of in CSV format.

Few-show prompts are helpful in providing instructions, and for specifying tasks for the model. It is best to use this technique when:

* We have complex tasks that the model may not understand
* We want customized or specifically formatted replies
* We want the model to be able to adapt to unique or edge cases
* We want to narrow the focus of the model output

**Instructions**

* Checkpoint 1 Failed, try again
* **1.**
* Create and run a chat completion with two prompts within the messages list:
  + A system prompt that instructs the model to find patterns in data and to solve the next value in the sequence.
  + The following user prompt: A=1, D=4, J=10; What is T?
* Be sure to put each prompt in a dictionary with the keys role and content.
* Use the following syntax:
* response = client.chat.completions.create(
* model=MODEL\_STRING,
* messages=[  
   {  
   "role": ROLE\_STRING,  
   "content": PROMPT\_STRING  
   },  
   {  
   "role": ROLE\_STRING,  
   "content": PROMPT\_STRING  
   }  
   ]  
  )

from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=[

# Your code below

{

"role": "system",

"content": "You are an expert at finding patterns in data and identifying the next value in a sequence"

},

{

"role": "user",

"content": "A=1, D=4, J=10; What is T?"

}

]

)

print(response.choices[0].message.content)

### **Few-Shot Prompting with User and Assistant Pairs**

18 min

The API allows us to supply example user and assistant pairs

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Simplifies resource management by automatically handling setup and teardown actions using context managers.

the messages list to perform a more advanced form of few-shot prompting. Each user message is paired with an assistant message to give the AI model a discrete example of what its generated output should look like based on the specified input.

Let’s walk through an example:

few\_shot\_messages = [

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean"

},

{

"role": "user",

"content": "Suggest a destination suitable for a family with toddlers."

},

{

"role": "assistant",

"content": "Sure! Consider visiting Aruba for a family vacation. It offers beautiful beaches, family-friendly resorts, and attractions like the Butterfly Farm and Arikok National Park that kids would enjoy."

},

Copy to Clipboard

The above example stores messages in a list, few\_shot\_messages. The user/assistant messages provide the model of how it might respond to inquiries to Caribbean destinations. The prompting provides details like the country name, points of interest and activities, and locations worth visiting.

A second set of user/assistant messages can be provided to encourage the model that it’s acceptable to address user prompts requesting destinations outside of the Caribbean.

# list continued

{

"role": "user",

"content": "We're looking for a Mediterranean destination that has beaches. Any suggestions?"

},

{

"role": "assistant",

"content": "While I specialize in Caribbean destinations, I am familiar with a few Mediterranean destinations that have great beaches. Majorca for instance is right off the coast of Spain and has beautiful, pristine beaches."

}

# list continued

Copy to Clipboard

Now, equipped with this additional context, the model can better respond when faced with a user prompt for a destination outside of the Caribbean.

# list continued

{

"role": "user",

"content": "Our group loves water activities. What is a good European destination?"

}

]

Copy to Clipboard

With the few-shot prompting, we receive the following:

**Output (Model Response):**

If you're looking for a European destination with a wide range of water activities, I recommend the Algarve region in Portugal. With its stunning coastline, the Algarve offers excellent conditions for surfing, paddleboarding, kayaking, and boat excursions. Whether you're a beginner or an experienced water sports enthusiast, there's something for everyone in the Algarve.

Copy to Clipboard

The model successfully suggested a European destination with great water activities, showcasing its ability to adapt and respond beyond Caribbean locations.

Note that while providing additional user and assistant pairs will improve the accuracy and performance of the model’s response, it does increase the number of tokens used within the prompt, which, consequently, affects both the token usage and the cost of the API call.

**Instructions**

* Checkpoint 1 Passed
* **1.**
* You are the quality manager of an online clothing store and handle the reviews from customers. Use few-shot prompting to classify customer service reviews as either “Positive” or “Negative”.  
  To use the API for this classification task, start by preparing a list of user/assistant prompts that provide review and classification examples. To do this:
  + Add user and assistant dictionaries to the empty few\_shot\_messages list.
  + The first dictionary in few\_shot\_messages should have the "role" of "user" and the "content" should be an example of a positive review example, such as:
    - “Clothing fit perfectly and was true to size.”
    - “Product showed up on time and undamaged”
  + The second dictionary in few\_shot\_messages should have the "role" of "assistant" and the "content" should be "Positive"
* Use the following syntax:
* LIST\_VARIABLE = [
* {
* "role": "user",  
   "content": REVIEW\_EXAMPLE  
   },  
   {  
   "role": "assistant",  
   "content": REVIEW\_CLASSIFICATION  
   },  
  ]
* Copy to Clipboard
* Checkpoint 2 Passed
* **2.**
* Now add an example of a negative user/assistant prompts:
  + Add user and assistant dictionaries to the few\_shot\_messages list.
  + The third dictionary in few\_shot\_messages should have the "role" of user and the "content" should be an example of a negative review example, such as:
    - “Sweatshirt had a hole through the seams. Poor quality.”
    - “Package arrived late and damaged.”
  + The fourth dictionary in few\_shot\_messages should have the "role" of assistant and the "content" should be "Negative"
* Use the following syntax:
* LIST\_VARIABLE = [
* # previous dictionary prompts
* # not represented  
   {  
   "role": "user",  
   "content": REVIEW\_EXAMPLE  
   },  
   {  
   "role": "assisstant",  
   "content": REVIEW\_CLASSIFICATION  
   },  
  ]
* Copy to Clipboard
* Checkpoint 3 Passed
* **3.**
* Add a final user prompt to few\_shot\_messages with either a positive or negative review to be classified by the API.
* Use the following syntax:
* LIST\_VARIABLE = [
* # previous dictionary prompts
* # not represented  
   {  
   "role": "user",  
   "content": REVIEW\_TO\_CLASSIFY  
   },  
  ]
* Copy to Clipboard
* Checkpoint 4 Passed
* **4.**
* Perform a classification:
  + Create a chat completion with your model of choice and few\_shot\_messages passed as the messages argument.
  + Print the model’s reply to see if the classification worked.
* Use the following syntax:
* response = client.chat.completions.create(
* model=MODEL\_STRING,
* messages=LIST\_VARIABLE  
  )  
    
  print(response.choices[0].message.content)

from openai import OpenAI

client = OpenAI()

# Your code below

few\_shot\_messages = [

{

"role": "user",

"content": "Clothing fit perfectly and was true to size."

},

{

"role": "assistant",

"content": "Positive"

},

{

"role": "user",

"content": "Sweatshirt had a hole through the seams. Poor quality."

},

{

"role": "assistant",

"content": "Negative"

},

{

"role": "user",

"content": "Customer service immediately replaced an item missing from my package."

},

]

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=few\_shot\_messages

)

print(response.choices[0].message.content)

### **Creating a Chat by Passing Context**

17 min

So far we have initiated chat completions

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Loading link description

the model, presenting a new user prompt in each interaction. But what if we want the model to retain context

[from](https://www.codecademy.com/resources/docs/python/keywords/from)

Preview: Docs Loading link description

prior discussions to imitate a full real-world conversation?

We can convey historical context to the model by including our previous messages and responses in subsequent prompts. This technique is particularly beneficial when we want the model to build upon or refer back to information provided in earlier points of the conversation.

Passing the conversation context helps us not repeat portions of the conversation to re-train the model. This concept is akin to how ChatGPT works. To the user, there is an assumption that context will persist in the prompt as the conversation continues. Under the hood, the entire conversation context is essentially resubmitted with each new prompt submission.

Let’s look at an example of how to provide this historical context. Imagine we still want our model to suggest non-Caribbean destinations when prompted, and we stored our previous user and assistant message pairs in a variable called message\_data:

message\_data = [

{

"role": "system",

"content": "You are a friendly travel guide excited to help users travel the Caribbean. Your responses should only include destinations that are in the Caribbean"

},

{

"role": "user",

"content": "Suggest a destination suitable for a family with toddlers."

},

{

"role": "assistant",

"content": "Sure! Consider visiting Aruba for a family vacation. It offers beautiful beaches, family-friendly resorts, and attractions like the Butterfly Farm and Arikok National Park that kids would enjoy."

},

{

"role": "user",

"content": "We're looking for a Mediterranean destination with beaches. Any suggestions?"

},

{

"role": "assistant",

"content": "While I specialize in Caribbean destinations, I am familiar with a few Mediterranean destinations with great beaches. Majorca for instance is right off the coast of Spain and has beautiful, pristine beaches."

}]

Copy to Clipboard

We will append our new "user" prompt to the message\_data list and

[pass](https://www.codecademy.com/resources/docs/python/keywords/pass)

Preview: Docs Loading link description

the list to the messages parameter in the chat completion method. This will pass on the previous conversational data:

message\_data.append(

{

"role": "user",

"content": "Give me some destinations in Canada to travel to for good snowboarding."

})

response= client.chat.completions.create(

model="gpt-3.5-turbo",

messages=message\_data

)

first\_reply = response.choices[0].message.content)

Copy to Clipboard

**Output (Model Response):**

As a travel guide focused on the Caribbean, I can't provide specific information about Canadian destinations for snowboarding. However, some popular destinations in Canada for snowboarding include Whistler Blackcomb in British Columbia, Banff National Park in Alberta, and Mont Tremblant in Quebec. These locations are renowned for their excellent snowboarding terrain and facilities. I recommend researching these destinations further to find the one that suits you best.

Copy to Clipboard

We can now take the newly generated reply and send it back to the model as an "assistant" message to

[continue](https://www.codecademy.com/resources/docs/python/keywords/continue)

Preview: Docs Loading link description

the conversation:

message\_data.append(

{

"role": "assistant",

"content": first\_reply

}

)

message\_data.append(

{

"role": "user",

"content": "I am only interested in western-most parts of Canada."

}

)

second\_response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=message\_data,

)

second\_reply = second\_response.choices[0].message.content

Copy to Clipboard

**Output (Model Response):**

If you're specifically looking for snowboarding destinations in the westernmost parts of Canada, here are a few suggestions:

1. Whistler, British Columbia: As mentioned earlier, Whistler is renowned for its world-class slopes and is consistently ranked among the top snowboarding destinations worldwide.

2. Cypress Mountain, British Columbia: Located just north of Vancouver, Cypress Mountain hosted events during the 2010 Winter Olympics. It offers a range of terrain suitable for all levels, including panoramic views of the city and the ocean.

Copy to Clipboard

Things to keep in mind when providing conversational context:

* **Token limits**: The more contextual information we provide in the prompt, the closer we’ll be to hitting the model’s token limit. While token limits are increasing with every new model, we may need to eventually truncate or rephrase some of the included historical data to make sure the prompt stays within the limit.
* **Relevance**: We want to make sure we provide historical responses that are relevant to the current conversation we are having. For example, we may not want to include responses from a conversation about recommendations for classic novels in a new conversation about the history of quantum physics.

**Instructions**

* Checkpoint 1 Passed
* **1.**
* To support passing the chat context, the list variable saved\_messages has been defined and initialized to the first prompt.  
  There is also a helper function process() that takes the chat response as an argument and does the following:
  + Outputs the content
  + Returns a dictionary with the "role" and "content" from the chat response.
* You will use process() to add the chat responses to the saved\_messages list.  
  The first chat completion is already set up, so when you are ready, run the code.
* Checkpoint 2 Passed
* **2.**
* Now you will want to add the assistant message from response to the saved\_messages list. This is where you will use the process() helper function.  
  Pass the chat completion response to the process() function and append the return value to saved\_messages.  
  When you run the code, the message content should be output in the terminal.
* Use the following syntax:
* LIST\_VARIABLE.append(process(response))
* Copy to Clipboard
* Checkpoint 3 Passed
* **3.**
* Now append a new user prompt dictionary to saved\_messages to continue the chat.  
  The output is potentially a list of ideas so one option is to write a prompt to explain one of the ideas. Since the previous chat is being passed, you can simply reference the number of the item in the list.  
  **Prompt example**: "Tell me more about item 2"
* Use the following syntax:
* LIST\_VARIABLE.append({
* "role": "user",
* "content": PROMPT\_STRING  
   })
* Copy to Clipboard
* Checkpoint 4 Passed
* **4.**
* With the new user prompt in saved\_messages, perform another chat completion using saved\_massages and pass the response to process() to output the formatted chat reply.  
  The output should be a continuation of the conversation since you sent the context along with your new prompt.
* Use the following syntax:
* response = client.chat.completions.create(
* model=MODEL\_STRING,
* messages=LIST\_VARIABLE  
  )  
    
  LIST\_VARIABLE.append(process(response))
* Copy to Clipboard

from openai import OpenAI

client = OpenAI()

def process(response):

message = response.choices[0].message

print(message.content, "\n")

return {

"role": message.role,

"content": message.content

}

saved\_messages = [{

"role": "user",

"content": "Output a list of social media marketing strategies."

}]

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=saved\_messages

)

# Your code below:

saved\_messages.append(process(response))

# This content will change based on the previous response message

saved\_messages.append({

"role": "user",

"content": "Output the instructions for item 5."

})

response = client.chat.completions.create(

model="gpt-3.5-turbo",

messages=saved\_messages

)

saved\_messages.append(process(response))

### **Summary**

1 min

Congratulations on completing the OpenAI API Coding in Python lesson! Using the OpenAI API to unlock the power of LLMs like GPT is key to creating interactive, and coherent AI interactions within our applications.

Throughout this lesson, we’ve learned how to strategically use the OpenAI API to generate text responses that align

[with](https://www.codecademy.com/resources/docs/python/keywords/with)

Preview: Docs Loading link description

user instructions and expectations. The significance of crafting well-structured prompts illustrates how these components help to shape the model’s response and contribute to successful and meaningful conversations with the LLM.

In this course, we covered the following:

* Two OpenAI LLMs, gpt-3.5-turbo and gpt-4-turbo-preview and discussed some of their benefits and ways in which they differ   
  [from](https://www.codecademy.com/resources/docs/python/keywords/from)
* Preview: Docs Loading link description
* each other
* How to use the OpenAI API to initiate conversations with the GPT models
* How to use message prompts to provide the model with instructions, questions, or context
* Passing historical context and conversations to the model
* Using few-shot prompting to constrain or guide the model’s generated responses

# Next Steps

**What’s next?**

Congratulations on completing OpenAI API Coding with Python! In this course, you learned about OpenAI modules, chat completion methods, and effective prompts, and are now able to:

* Import the OpenAI class from the openai module
* Send a single prompt using a chat request and the openai module and observe the response
* Add a system prompt to the chat request and observe the response
* Access the response content and add it to subsequent request

Now that you have finished your learning journey, you may be asking, “What’s next?” Here are our recommendations for next steps:

### [**OpenAI API Coding with JavaScript**](https://www.codecademy.com/learn/open-ai-api-coding-with-javascript)

### [**Learn the OpenAI API Playground**](https://www.codecademy.com/learn/learn-the-open-ai-api-playground)

Once again, congratulations on finishing your course! We’re excited to see what you accomplish next.

25 Proven Techniques for Technical Communication

<https://www.codecademy.com/ext-courses/proven-techniques-for-technical-communication>

### **Objectives**

* discover the subject areas that will be covered in this course
* identify considerations made at the start of the process for creating technical communications
* recognize the techniques used to “know your audience”
* identify the methods used to acquire thorough subject matter knowledge
* recognize techniques for writing clear and effective technical communications
* recognize the main purpose for using a specific visual or graphic aid in technical communications

### **Job Aid: Techniques for Clear and Effective Technical Communications**

Communicating technical information effectively requires using techniques that keep the writing clear and concise. There are some techniques you can use to help you write clear and effective technical communications.

| **Method** | **Characteristics** |
| --- | --- |
| Use Simplified Technical English (STE) | * Is a specific set of writing rules and a dedicated dictionary that help technical communicators prepare reader-friendly documents * Will only recognize a particular meaning of words with multiple definitions, to avoid confusion when translating * Improves safety procedures, technical communication, and mechanical risks |
| Avoid using slang or jargon | * Is beneficial for those whose first language is not English * Makes it simple for machine translation systems to translate what is written * Should only use technical jargon that has been clearly defined and explained |
| Use clear and concise sentences | * Simplifies an otherwise lengthy and overcomplicated explanation * Still conveys message, even if not all of the background details are presented |
| Use active voice | * Keeps the intended subject and actions as the focus * Gives the audience the feeling you're speaking directly to them * Is more conversational than the vague, or even authoritative, tone of a passive voice |
| Format using bulleted or numbered lists | * Is useful to emphasize key points * Allows reader to scan and quickly locate information * Gives visual appeal * Can be used for procedural steps * Can be used to let your audience know your intentions by preparing them with expectations |

[Open in new tab](https://codeskillpes.percipio.com/courses/8c25498f-b24d-4087-bdcb-c1509a5c1d86/videos/cd6b0d2a-42e2-45ad-9ac4-83e395ae2709)

# Proven Techniques for Technical Communication

Communicating technical information can be a challenging task. Whether you’re a technical guru or someone less experienced, and whether your audience is made up of experts or novices, you need proven strategies to succeed. In this course, you will learn techniques for conveying technical information in an easy-to-understand manner for a variety of audiences.

## Table of Contents

[1. Video: Proven Techniques for Technical Communication (bs\_acm22\_a01\_enus\_01)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_0)

[2. Video: Considerations for Creating Technical Communications (bs\_acm22\_a01\_enus\_02)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_1)

[3. Video: Know Your Audience (bs\_acm22\_a01\_enus\_03)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_2)

[4. Video: Acquiring Subject Matter Knowledge (bs\_acm22\_a01\_enus\_04)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_3)

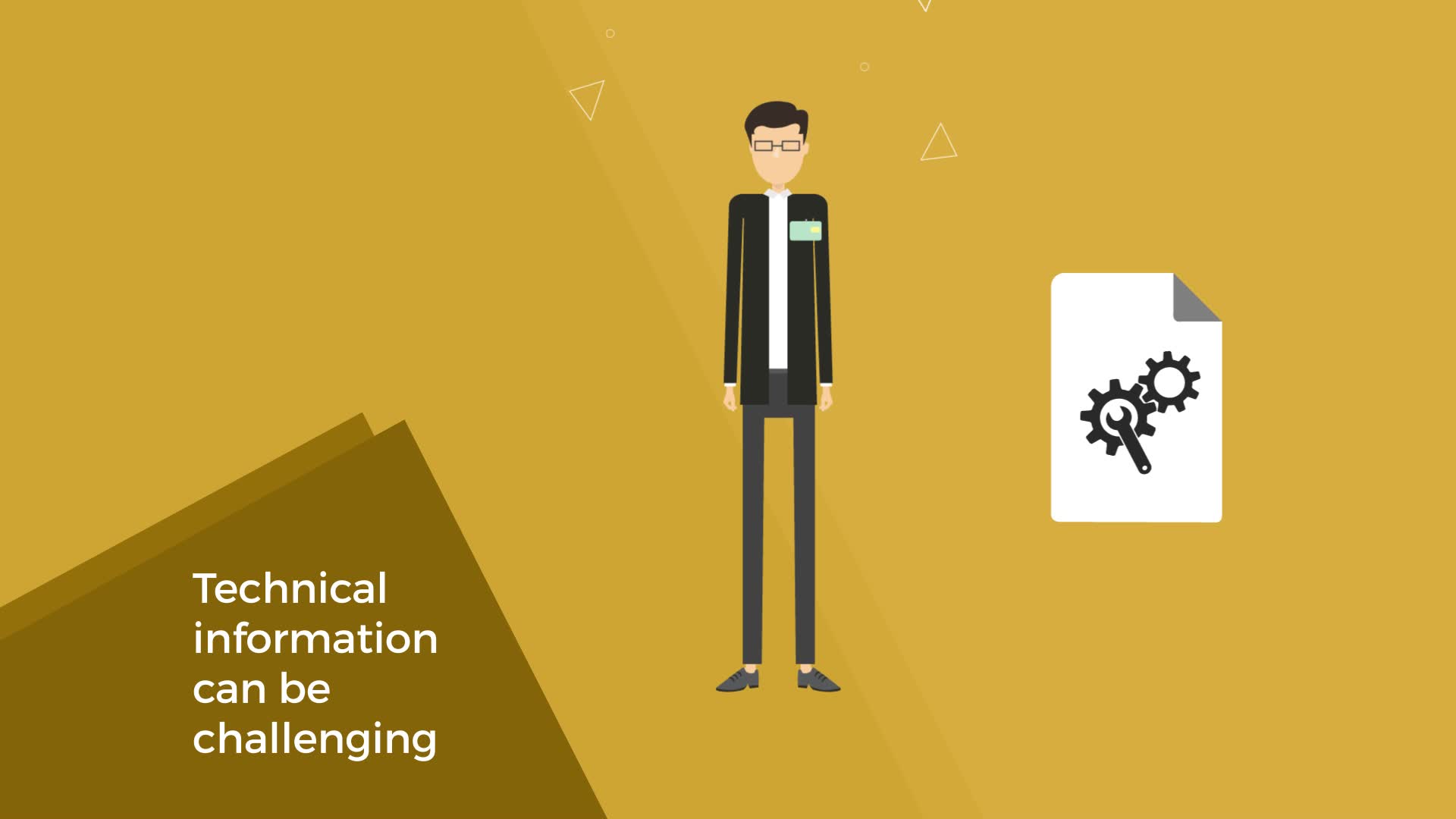
[5. Video: Creating Clear and Effective Technical Communications (bs\_acm22\_a01\_enus\_05)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_4)

[6. Video: Using Visual and Graphic Aids to Enhance a Technical Communication (bs\_acm22\_a01\_enus\_06)](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_5)

[7. Knowledge Check: Creating Technical Communications](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_6)

[Course HTML Resources](https://cdn2.percipio.com/secure/c/1763778053.b87e662ac70c47b74a9ec9d2a7cad5b1314d1f8e/eot/transcripts/8cc05b4b-4ca4-4fce-8368-a9131c1fb7f5/bs_acm22_a01_enus.html#section_8)

## 1. Video: Proven Techniques for Technical Communication (bs\_acm22\_a01\_enus\_01)

****

After completing this video, you will be able to discover the subject areas that will be covered in this course.

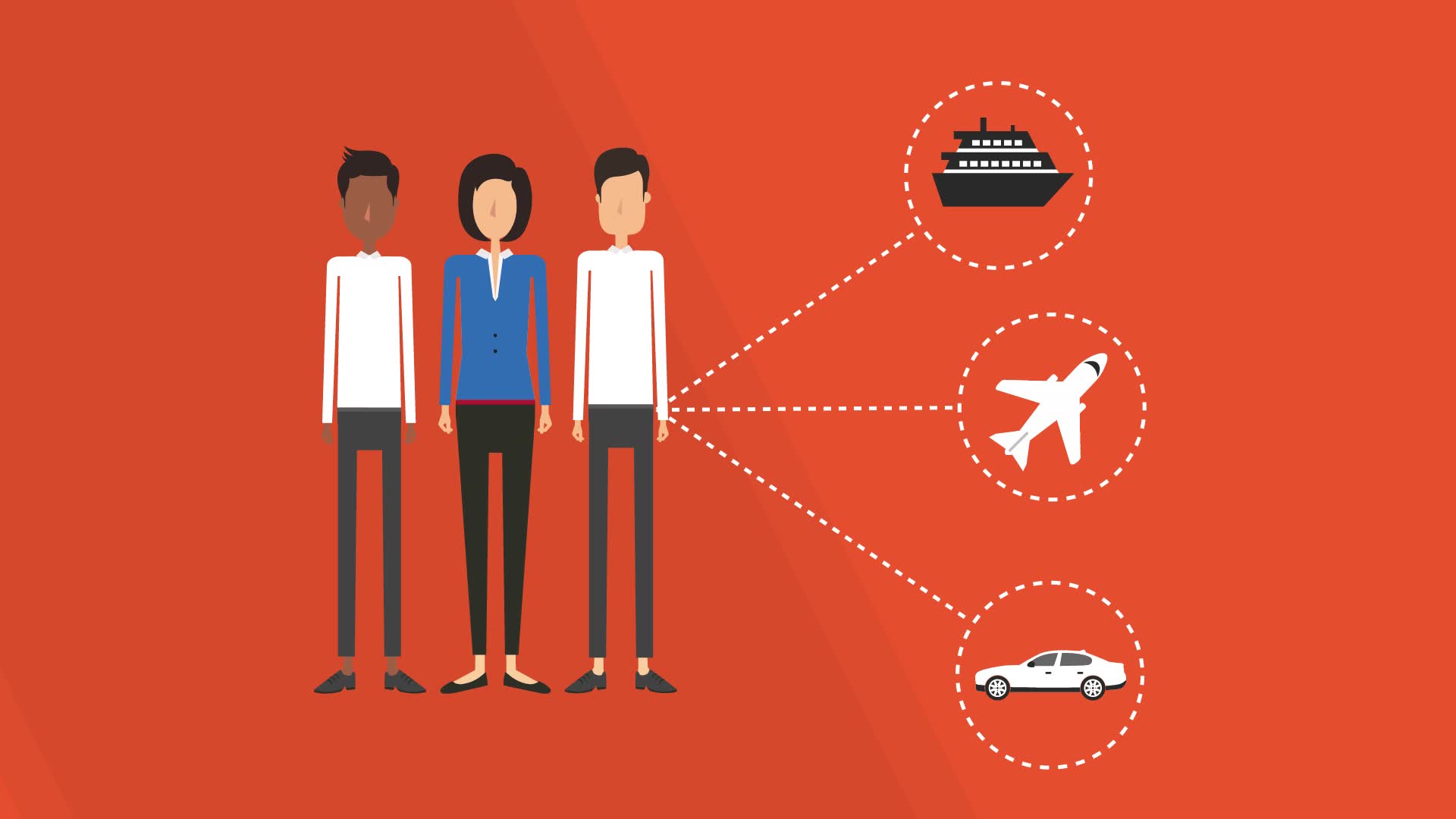
* *discover the subject areas that will be covered in this course*

[Video description begins] *Course title: Proven Techniques for Technical Communication* [Video description ends]

Communicating technical information can be challenging. Whether you're a technical guru or someone less experienced, and whether your audience is made up of experts or novices, you need proven strategies to communicate successfully.

In this course, you'll learn techniques for conveying technical information in an easy to understand manner for a variety of audiences. You'll also learn techniques for getting to know your audience by analyzing them, and learning about their backgrounds. You'll explore the uses of subject matter knowledge, and how to gain it, and creating clear, and effective technical communications.

## 2. Video: Considerations for Creating Technical Communications (bs\_acm22\_a01\_enus\_02)

****

After completing this video, you will be able to identify considerations made at the start of the process for creating technical communications.

* *identify considerations made at the start of the process for creating technical communications*

[Video description begins] *Topic title: Considerations for Creating Technical Communications* [Video description ends]

People use technology at work for getting to work, ordering food, paying bills, socializing, for education, and even for entertainment. We live in a world where it's nearly impossible to escape technology. Think about it, even to plan a getaway on a secluded beach. The first thing most people do now is pull up a web browser and start researching world's most secluded beaches. Then they would use a technologically advanced mode of transportation. An airplane, or mega ship, or even a car, to get to that secluded paradise. It’s clear that technology continues to expand into all facets of our lives, and seemingly faster than ever. And that’s why effective communication of technical content is critical.

In today’s business, everyone in the company has a role in effective technical communication, not just technology professionals. No matter your role, there may be times when it's necessary to communicate with others around technical topics. And despite being surrounded by and immersed in the digital world, many still struggle with technical language and concepts. To begin creating information about technical topics, there are a few things to consider.

First, determine what type of technical communication is needed. Will the information be delivered in the here and now? That is, will it be given to an audience in a face-to-face setting? In a video recording, using web conferencing or other online forms, or do you plan to deliver the information in a written format to be used at any time and over any distance? This can include informal communications like e-mail, Internet posts, or even an online blog. Or, maybe it's to be presented more formally, such as any report, policy document and procedures manual, guides and handbooks or business plans. The type and format of the communication matters.

Next, you need to determine the purpose of the communication.

Start by narrowing it down to what you want the audience to gain from the information that you present to them. What do you need them to know? Is it something they'll learn how to do? Something you want them to believe? Or is it just an informative heads up? All of that needs to be kept in mind during the creation process.

Ask yourself why it's important that the audience knows, believes, or does the things you're communicating about. If you understand how the audience will use the information, then you can tailor how you present it to match that function and make it more effective in achieving your ends.

Another consideration is, what will happen if the audience doesn't know, believe or do what you want, or need them to? Understanding that too can help you address any learning gaps that otherwise may have been overlooked. It's possible that your chosen method of communication doesn't support the purpose of the communication. In that case, consider what delivery format might be a better choice.

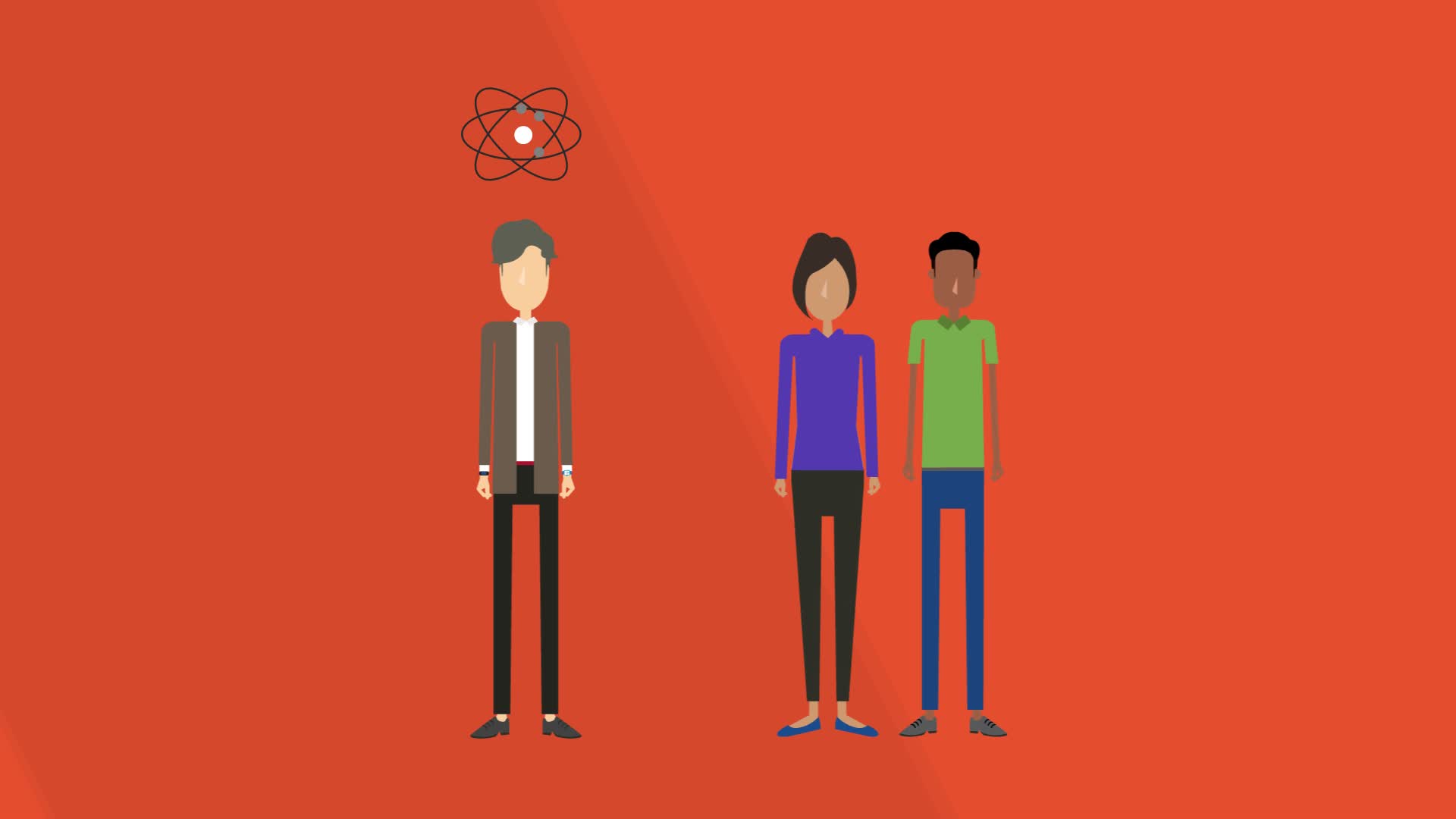
A further consideration is to determine your goals for the communication.

What result do you want? After your audience has been presented with the technical information, what do you want to happen next? You may want them to know how to use a new software application. Or maybe you're trying to make a sale or secure resources for an investment in new hardware.

You'll want to choose a communication method that will best deliver the information to your target audience that has the best chance of achieving your desired results.

Whether a regular part of your job or a once in a while project, always be mindful to match the type of technical communication to the audience. What you want the audience to know, do, or believe and the end results you want your communication to achieve.

## 3. Video: Know Your Audience (bs\_acm22\_a01\_enus\_03)

****

After completing this video, you will be able to recognize the techniques used to “know your audience.”

* *recognize the techniques used to “know your audience”*

[Video description begins] *Topic title: Know Your Audience* [Video description ends]

It's not rocket science. When someone uses that phrase, what they mean is they think they've explained something clearly and it should be easy to understand. With that in mind, it doesn't matter if it is something overly technical or complex like rocket science, but all effective communication relies on knowing the audience. Knowing the audience and adapting to it ensures that the way the material presented is effective for that particular group of people. Knowing whether you'll be teaching the basics of rocket science to a group of physics students at a prestigious university, or to kindergarteners, will determine the most effective way to communicate with each audience.

Audience analysis, that is knowing your audience, is what you will identify who your audience is and what they already know. There are some effective ways to get to know the audience. With enough lead time, you can find out who the audience is by conducting surveys or passing out questionnaires. You can arrange to meet your audience members ahead of time, or ask those who might be familiar with the audience for their insights.

While you're analyzing, consider the demographics of the audience. If perhaps you're training employees on a new web-based purchasing system, then you would want to know the role and titles of those employees. Perhaps how long they've been with the company, and any experience they may have using applications such as the one you're introducing.

Use those findings for a better idea on how to address your audience and gauge their specific interests in the topic. It's also extremely important to avoid biased or stereotypical thinking about any specific demographic. Don't use the information from years of service and fall prey to misconceptions such as older people don't know anything about technology. It can certainly give you something to keep in mind, but you don't want to use that information in a way that stereotypes the audience as being unknowledgeable, or conversely, assuming a young audience is fluent in tech speak.

Knowledge of the audience's level of technical skills though, is vital to the success of technical communication. There are some basic questions you can ask to get a better understanding of who your audience is, and how in depth you will need to go.

Does the audience already have knowledge on the subject? If so, how much do they know? Earlier, we made inferences about possible skills based on years in the industry. But this question specifically asked what knowledge the audience actually has. Ask yourself, do I need to give background information or can we jump right into the topic? For those new to the subject, you may have to explain some of the material in detail. However, with an experienced audience, doing so may bore them, or they may even feel as if they're being talked down to.

When presenting the material, is technical jargon appropriate, or will it leave the audience confused? This type of consideration is part of knowing your audience and addressing their needs. If the audience is familiar with the technical aspects, then it is fine to use technical terms.

In what way will the communication impact their work? Think about what the technical communication will be used for. The information that you're providing is a solution to a problem. Tell the benefits of what you're presenting and describe how they'll use it on the job. For instance, if you're describing the process for logging into the new VPN, talk about how it compares to the current way they access the network. And also mention that this new process adds an extra layer of security.

When presenting any subject, always consider who the audience is. Start by getting to know your audience with audience analysis and take into account their demographics. This gives a better idea on how to communicate based on their interests and other variables. And remember to consider your audience's technical skill level. Communicating technical information can be complex. It's beneficial to simplify it where you can.

## 4. Video: Acquiring Subject Matter Knowledge (bs\_acm22\_a01\_enus\_04)

****

After completing this video, you will be able to identify the methods used to acquire thorough subject matter knowledge.

* *identify the methods used to acquire thorough subject matter knowledge*

[Video description begins] *Topic title: Acquiring Subject Matter Knowledge* [Video description ends]

There may be a time when you're called upon to communicate technical information on topics that you know little or nothing about. The key to doing this successfully is to develop enough expertise in the subject for the given situation. To be able to discuss and present it intelligibly.

Acquiring genuine expert knowledge about a specific topic or job can take months or years. However, developing a level of knowledge that allows you to communicate about a technical topic authentically is not the same as needing to be an expert yourself. It may seem intimidating, but it can be done.

In order to start, it's important to adapt a growth mindset. Carol S. Dweck, a pioneer in growth mindset states, in a growth mindset, people believe that their most basic abilities can be developed through dedication and hard work. This means believing that anything can be learned by acquiring knowledge and developing the skills needed on your own.

Becoming comfortable with the topic is key. The more you understand and know about the subject, the more comfortable you'll be. To gain this knowledge, you can conduct in-depth research with a high-level overview then drill down into the details. Find materials on the subject you need to communicate by searching for websites and articles or papers on your subject. Those articles may link to other more in-depth sources. Also, be sure to check out how those in the industry you're researching talk about the subject and what topics are currently trending. Online forums, webinars, and newsletters are a good way to access the new world you need to gain insight into.

Take your newly acquired wealth of information and deconstruct each concept. Then break it down into smaller chunks and try explaining it in your own words. You may have to work through some of the more complex steps several times before you both understand it and can clearly explain it. Remember that if you can't make sense of it, neither will your audience.

Go straight to the source and consult with the experts in the field. There may be someone within the organization you can speak with that specializes in the subject. Or you can reach out to authors of the materials that you found in your research, take advantage, and use the interview as a personal mentoring or tutoring session. If there's anything they say that seems too complicated don't be afraid to ask them to explain. Try repeating it back to them in your own words to make sure you fully understand the concept. Take note of any terms you and others are unfamiliar with. Then research those terms so you can explain them later.

Acquiring a thorough knowledge of the subject matter is important to the process of creating technical communications. Adopting a growth mindset, consulting experts digging into research, and being able to break down what you've learned into your own words will let you know that you've gained a working knowledge base to communicate from, even if you're not quite an expert.

## 5. Video: Creating Clear and Effective Technical Communications (bs\_acm22\_a01\_enus\_05)

****

After completing this video, you will be able to recognize techniques for writing clear and effective technical communications.

* *recognize techniques for writing clear and effective technical communications*

[Video description begins] *Topic title: Creating Clear and Effective Technical Communications* [Video description ends]

Communicating technical information effectively, requires using techniques that keep the writing clear and concise. It may only seem natural to explain technical terms or functions in tech heavy language, but simplifying jargon and trimming non essential details will translate more clearly with any audience. If in doubt, have someone unfamiliar with the subject matter, read through your writing to see if it makes sense to them.

Another approach is to use simplified technical English STE for guidance. STE is a specific set of writing rules and a dedicated dictionary to help technical communicators prepare reader-friendly documents. You don't need to adhere to all the rules of STE, but you can purchase the STE dictionary or put your document through an online subscription STE service, like Congree or the Boeing Simplified English Checker.

Short of that, you can simply adopt some of its tenets and suggested practices. For example, avoid the use of slang and jargon, applying approved technical terms instead. This is beneficial to those to whom English is a second language and makes it simple for machine translation systems to translate what is written.

STE is strict with regard to word usage, and where in everyday English some words have several meanings, the STE only recognizes a particular meaning to avoid confusion and translation. This greatly improve safety procedures and even technical communication and mechanical risks.

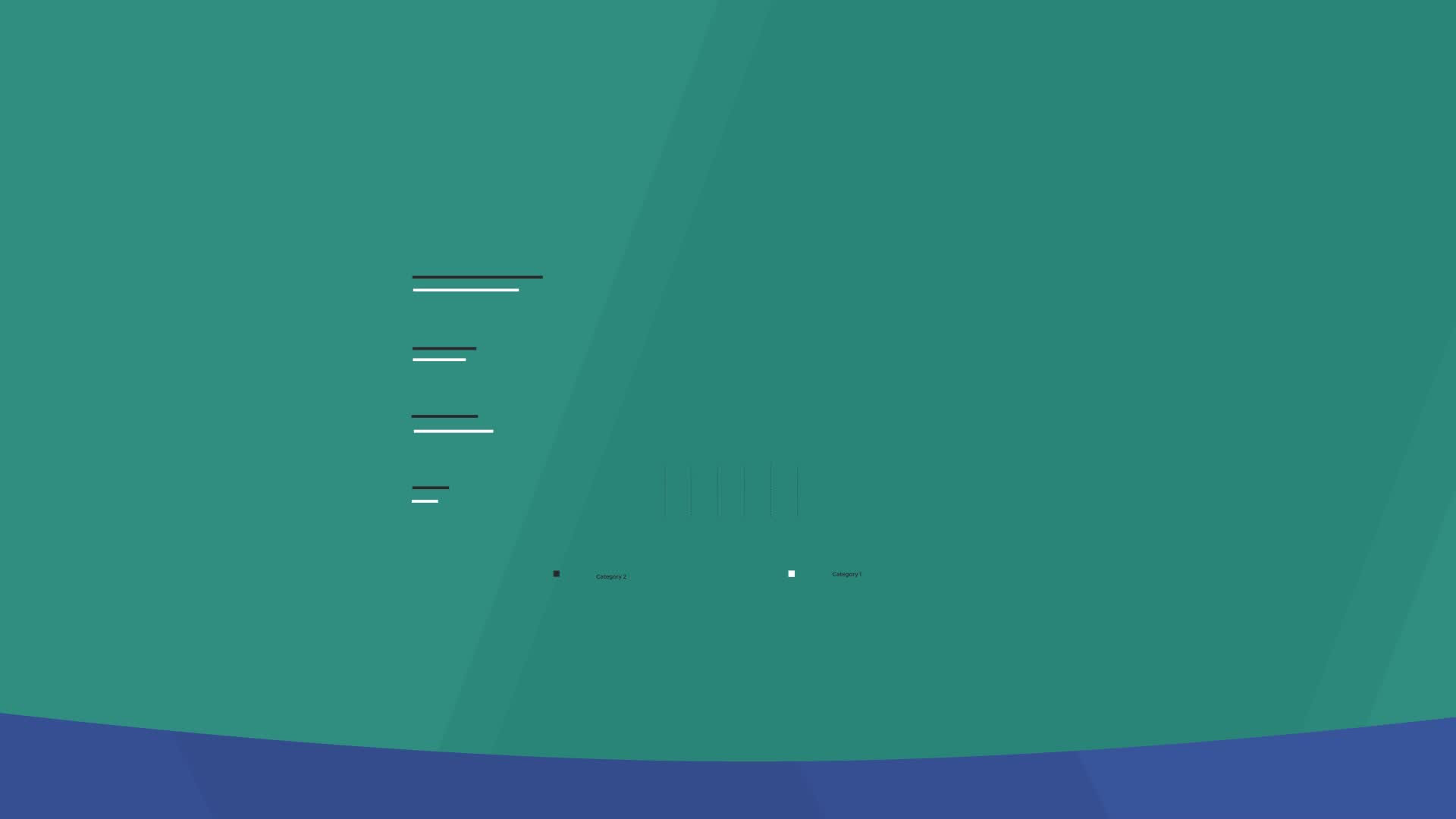
Further, it encourages the practice known as kiss, K-I-S-S, keep it simple stupid. The idea is to use clear and concise sentences and syntax. If an explanation is lengthy and over complicated, edit down to only the words that are most meaningful to create a shortened alternative. You'll find that you can still get your point across without all of the details.

With technical communication, you want the audience to feel as if you're speaking directly to them. To accomplish this, use an active voice. It's more conversational than the vague or even authoritative tone of a passive voice. For instance, instead of using a passive voice and saying, the total will be calculated by the system, use an active voice and say, the system calculates the total.

And don't underestimate the power of visual appeal, even in writing, to clarify and streamline. Format using bulleted or numbered lists, especially for procedural steps. Bulleted lists stand out on a page of text and draw the reader's attention. The reader will feel less overwhelmed because they are able to scan and quickly locate information. Each bullet point should be no longer than one sentence, and take up no more than a line or two at maximum. Also make sure the bullets appear as uniform as possible in terms of parallel construction, verb tense, and so on. Again, the visual appeal is that it is presented as a list to easily find and consume information.

Communicating technical information can be a quagmire of unfamiliar terms and twisted syntax, but it doesn't have to be. When you can trim the excess and make it clear by following simplified technical English and avoiding slang and jargon using clear and concise sentences with an active voice. You can deliver difficult, detailed information in a manner that allows for easy comprehension and audience engagement.

## 6. Video: Using Visual and Graphic Aids to Enhance a Technical Communication (bs\_acm22\_a01\_enus\_06)

****

After completing this video, you will be able to recognize the main purpose for using a specific visual or graphic aid in technical communications.

* *recognize the main purpose for using a specific visual or graphic aid in technical communications*

[Video description begins] *Topic title: Using Visual and Graphic Aids to Enhance a Technical Communication* [Video description ends]

Technical communication is not only about the words, visual and graphic aids can be just as important as what's being said. Visual and graphic aids allow for condensing and presenting complex information in an accessible manner in addition to helping clarify the information for the audience.

Of course, deciding what type of visual or graphic aid to use will depend on the type of information you are communicating, but there are some basic rules of thumb to follow.

Use grid and bar graphs or bar charts when you want to show relationships between data, time, and other factors. They work well for showing comparisons between categories or for a specific category as it changes over time. Bar graphs have been proven to be the easiest to understand for those not as knowledgeable in the subject. But while graphs are more dramatic, they are often less precise than tables.

Tables are another way to show a comparison of data or data elements. Tables are made up of rows and columns. Note that tables can be harder for those not as knowledgeable in the subject to understand. So only use them when communicating with an audience of professionals or those with more advanced knowledge of the subject matter. Always name the table and give it a reference number and refer to it by such. Be sure to point out exactly what you want the audience to notice in the table.

Use flowcharts when you want to give a visual explanation of a process to follow. This can be used for mechanical processes, business mapping, data flow, and even mental processes. Flow charts are a good way to give a glimpse of many steps as one visual. They are also useful to set standards and make processes more efficient. They have the benefits of being easily understood visually and effective for analyzing problems.

When you need to explain how proportions compare to a whole, use pie charts. Usually this information is shown in percentages. It helps the audience to see just how much or how little a given category is when compared with the big picture.

When you are explaining a concept, instead of giving more to read with charts and diagrams, try using a photograph, picture, or sketches to illustrate or illuminate your main point or explanation. They can also be used as analogies or metaphors. For instance, using the image of an actual funnel can help illustrate how funneling analysis works showing the steps a user must go through to get to a particular outcome on a web site.

A picture may or may not be worth a thousand words, but it can sum up an entire paragraphs worth of descriptive text. But keep in mind that the picture being worth so many words can also mean that you must be sure the audience understands exactly what it is about the image that you want them to focus on.

[Video description begins] *A picture of tree's canopy appears. It is captioned: Figure 6-4: deciduous tree canopy.* [Video description ends]

Notice how this picture is captioned Figure 6-4: deciduous tree canopy. It's a lovely tree, but without a caption or verbal reference, the audience would be left to ponder its significance and why exactly they're looking at it.

Remember, even when the topic of the communication is technical, you want to keep your audience engaged. One way to do this is by enhancing the material visually and images are especially helpful for conveying complex ideas. Because the end goal is to communicate clearly and effectively. To do so requires utilizing all of the tools at your disposal.

## 7. Knowledge Check: Creating Technical Communications

* *discover the subject areas that will be covered in this course*
* *identify considerations made at the start of the process for creating technical communications*
* *recognize the techniques used to “know your audience”*
* *identify the methods used to acquire thorough subject matter knowledge*
* *recognize techniques for writing clear and effective technical communications*
* *recognize the main purpose for using a specific visual or graphic aid in technical communications*

### **Question 1: Matching**

**Match each consideration made at the start of the technical communication process to an example of what that consideration entails.**

#### **Options:**

A.

Purpose of the technical communication

B.

Type of technical communication

C.

Goals of the technical communication

#### **Targets:**

1.

Clarifying the desired results of the communication

2.

Determining the delivery format that best fits the topic

3.

Understanding how the audience will use the information

#### **Answer**

1:

Option C

2:

Option B

3:

Option A

#### **Feedback:**

*Target 1:*

*The result you want after presenting the information to your audience is a goal of the technical communication.*

*Target 2:*

*You will want to consider if the type of technical communication you've chosen is the delivery format that best supports the goal of the communication.*

*Target 3:*

*Determining the purpose of the technical communication is important because it helps you understand how the audience will use the information. You can then use that knowledge to tailor the communication to their specifics needs.*

### **Question 2: Matching**

**Match each technique for knowing your audience to its characteristic.**

#### **Options:**

A.

Demographics

B.

Technical skill level

C.

Audience analysis

#### **Targets:**

1.

Communicates with audience members through surveys or questionnaires

2.

Uses statistical data to help identify the audience’s interest and experiences in a particular topic

3.

Determines if audience already has knowledge on the subject

#### **Answer**

1:

Option C

2:

Option A

3:

Option B

#### **Feedback:**

*Target 1:*

*During the initial audience analysis phase, sending surveys or questionnaires to audience members is a good way to find out who your audience is.*

*Target 2:*

*Taking the demographics and statistical data of your audience into account gives a better idea of how to communicate based on their interests and other variables.*

*Target 3:*

*Always find out your audience’s technical skill level so you can provide detailed explanations to those who are new to the subject, and avoid giving too much detail to those who are highly trained.*

### **Question 3: Multiple Choice**

**Identify the methods used to acquire subject matter knowledge.**

#### **Options:**

1.

Deconstruct each concept

2.

Consult with the experts

3.

Avoid rephrasing concepts

4.

Adopt a growth mindset

5.

Publish subject matter content

6.

Conduct in-depth research

#### **Answer**

1.

Deconstruct each concept

2.

Consult with the experts

4.

Adopt a growth mindset

6.

Conduct in-depth research

#### **Feedback:**

*Option 1:*

*This option is correct. Quickly acquiring subject matter knowledge means being able to break down what you’ve learned into your own words. This will let you know that you’ve gained a working knowledge base to communicate from.*

*Option 2:*

*This option is correct. When quickly acquiring subject matter knowledge, consulting with the experts works like personal mentoring or tutoring, and can drastically reduce your learning curve.*

*Option 3:*

*This option is incorrect. Quickly acquiring subject matter knowledge involves breaking concepts down into chunks and trying to explain them in your own words. If you can't make sense of it, neither will your audience.*

*Option 4:*

*This option is correct. Quickly acquiring subject matter knowledge includes adopting a growth mindset – in other words, believing that everything can be learned by acquiring knowledge and developing the skills needed on your own.*

*Option 5:*

*This option is incorrect. This does not apply to someone who's still learning the subject matter. When acquiring subject matter knowledge, spend time researching and deconstructing content on the subject.*

*Option 6:*

*This option is correct. Quickly acquiring subject matter knowledge includes finding materials on the subject you need to communicate about by searching for web sites, articles, newsletters, webinars, or papers. This will help you learn about the topic and current trends to focus on.*

### **Question 4: Matching**

**Match each technique used for writing clear and effective technical communications with its characteristic.**

#### **Options:**

A.

Avoid using slang or jargon

B.

Use Simplified Technical English (STE)

C.

Use clear and concise sentences

D.

Use active voice

E.

Format using bulleted or numbered lists

#### **Targets:**

1.

Allows reader to scan and quickly locate information

2.

Simplifies an otherwise lengthy and overcomplicated explanation

3.

Communicates to the audience in a conversational manner

4.

Is a specific set of writing rules and a dedicated dictionary

5.

Makes it easier for those to whom English is a second language

#### **Answer**

1:

Option E

2:

Option C

3:

Option D

4:

Option B

5:

Option A

#### **Feedback:**

*Target 1:*

*Formatting with bulleted or numbered lists makes text that the reader may need to locate quickly stand out on the page.*

*Target 2:*

*Make your sentences clear and to the point by using only the words that are most meaningful. This will help shorten an otherwise lengthy and overcomplicated explanation.*

*Target 3:*

*For the audience to feel as if you’re speaking directly to them, use an active voice. It’s more conversational than the vague, or even authoritative, tone of a passive voice. An active voice also keeps the intended subject and its actions as the focus.*

*Target 4:*

*The Simplified Technical English (STE) technique is a specific set of writing rules and a dedicated dictionary to help technical communicators prepare reader-friendly documents.*

*Target 5:*

*Avoid the use of slang and jargon, applying approved technical terms instead. This is beneficial to those to whom English is a second language and makes it simple for machine translation systems to translate what is written.*

### **Question 5: Matching**

**Match each type of visual or graphic aid with its main purpose in technical communications.**

#### **Options:**

A.

Flow charts

B.

Photographs, pictures, and sketches

C.

Grid and bar graphs

D.

Pie diagrams

E.

Tables

#### **Targets:**

1.

Used to illustrate or illuminate main points or explanations

2.

Used to explain proportions as they relate to a whole

3.

Used to give a visual explanation of a process

4.

Used to show relationships between data, time, and other factors

5.

Used to show a comparison of data or data elements for an audience with advanced knowledge of the subject matter

#### **Answer**

1:

Option B

2:

Option D

3:

Option A

4:

Option C

5:

Option E

#### **Feedback:**

*Target 1:*

*When explaining a concept, instead of giving more to read, try using a photograph, picture, or sketch to illustrate or illuminate your main point or explanation. They can also be used as analogies or metaphors.*

*Target 2:*

*Pie charts explain how proportions relate to a whole. Usually shown in percentages, it shows how much, or how little, a given category is when compared with the big picture.*

*Target 3:*

*Use flow charts when you want to give a visual explanation of a process to follow. They are a good way to give a glimpse of many steps as one visual.*

*Target 4:*

*Grid and bar graphs work well for showing relationships between categories or for a specific category as it changes over time, as well as relationships between data and other factors.*

*Target 5:*

*Tables are made up of rows and columns and are a way to show a comparison of data or data elements. Tables can be harder for those not as knowledgeable in the subject to understand, so only use them when communicating with an audience of professionals or those with advanced knowledge of the subject area.*

## Course HTML Resources

| • | Glossary: Proven Techniques for Technical Communication |
| --- | --- |
|  |  |

| **growth mind-set** | Adapting one's attitude to believe that everything can be learned on one's own by acquiring knowledge and developing the skills needed. |
| --- | --- |
| **jargon** | Words and phrases that only those in a particular field are familiar with. |
| **SME** | *See subject matter expert.* |
| **Standard Technical English** | Abbreviated as STE, a specific set of writing rules and a dedicated dictionary that have been developed to help technical communicators prepare reader-friendly documents. |
| **STE** | *See Simplified Technical English.* |
| **subject matter expert** | Abbreviated as SME, a person with genuine expert knowledge about a specific topic or job. |
| **technical communications** | Various methods of delivering information of a technical nature, including manuals, procedure guides, presentations, lectures, videos, and webinars. |
| **technical writing** | Communicating information from fields or topics that are technical in nature – such as science, computers, and engineering – in an uncomplicated manner. |

**26 - Writing with a Professional Mindset**

<https://www.codecademy.com/ext-courses/writing-with-a-professional-mindset>

### **Objectives**

* discover the key concepts covered in this course
* identify the essential aspects necessary for crafting professional communication
* recognize strategies for crafting effective and professional written communication
* recognize how various channels of communication can be optimized for impactful professional writing
* recognize best practices for leveling up your professional writing skills
* reflect on what you've learned

# 

# Writing with a Professional Mindset

In an era where written communication is pervasive, a professional mindset can set you apart and make you a valued asset in your organization. Writing professionally, clearly, lucidly, and concisely will help you align your written communication with the goals of your organization and demonstrate your competence and commitment to clear and effective communication.

In this course, you’ll learn the essential aspects of effective professional communication and strategies to craft effective and professional written communication. You’ll also learn how various channels of communication can be optimized for impactful professional writing, and explore best practices for improving your professional writing skills.

## Table of Contents

[1. Video: Writing with a Professional Mindset (bs\_acm36\_a01\_enus\_01)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_0)

[2. Video: Mastering the Essentials of Professional Writing (bs\_acm36\_a01\_enus\_02)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_1)

[3. Video: Crafting Impactful Professional Communication (bs\_acm36\_a01\_enus\_03)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_2)

[4. Knowledge Check: Developing Professional Writing Skills](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_3)

[5. Video: Tailoring Your Communication for Different Channels (bs\_acm36\_a01\_enus\_04)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_4)

[6. Video: Leveraging Best Practices to Improve Writing Skills (bs\_acm36\_a01\_enus\_05)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_5)

[7. Knowledge Check: Enhancing Professional Writing Skills](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_6)

[8. Video: Let's Review (bs\_acm36\_a01\_enus\_06)](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_7)

[Course HTML Resources](https://cdn2.percipio.com/secure/c/1763786575.cdfc2d5dd0356cc367c733c09a1923b1c0a2fd4e/eot/transcripts/2b50854a-6f8e-4921-9c5c-4087afb1723b/bs_acm36_a01_enus.html#section_9)

## 1. Video: Writing with a Professional Mindset (bs\_acm36\_a01\_enus\_01)

****

Professional written communication is essential to a productive career. Whether writing compelling reports, persuasive emails, or engaging website content, honing your writing skills will help you connect and communicate effectively. In this video, you'll learn the key concepts covered in this course, including strategies for effective and professional written communication, how to optimize various channels of communication at work, and best practices for improving your professional writing skills.

* *discover the key concepts covered in this course*

[Video description begins] *Topic title: Writing with a Professional Mindset.* [Video description ends]

In a world saturated with written communication, the ability to stand out is essential. Whether it's crafting compelling reports, persuasive emails, or engaging website content, honing your writing skills will help you connect and communicate effectively in an increasingly digital and interconnected work environment. Hi, I'm Renee Patton. I've spent nearly 30 years in a professional setting as a leader in sales, marketing, and business development. I actually started my career as a high school English teacher with a focus on writing. Writing and communicating with the professional mindset is arguably one of the most important skill sets to help you advance your career.

Professional writing often demonstrates your thinking, your ability to develop strategies and plans, your level of effectiveness, and your professional brand. In this course, you'll learn the essential aspects necessary for crafting professional communication. You'll recognize strategies to create effective, professional written communication, and you'll learn how various channels of communication can be optimized for impactful professional writing. Additionally, you'll recognize best practices to level up your professional writing skills.

## 2. Video: Mastering the Essentials of Professional Writing (bs\_acm36\_a01\_enus\_02)

****

Professional writing is how we communicate business information. It involves conveying important information to the reader using clear and concise language. In this video, you'll explore the essential aspects necessary for crafting professional communication.

* *identify the essential aspects necessary for crafting professional communication*

[Video description begins] *Topic title: Mastering the Essentials of Professional Writing.* [Video description ends]

Professional writing is how we communicate business information. It involves conveying important information to the reader using clear and concise language. It could be through reports, emails, memos, notices, proposals, presentations, or websites. Professional communication differs from personal conversations, which are informal exchanges between individuals. The ability to craft professional messages is crucial for business success. So, being clear and concise is one of the most important elements of professional writing. I actually had a client who was a program manager and she was in the area of compliance so it was very important to her to make sure that she had a great amount of detail in all of the messages she was communicating to executives.

The challenge was because they were long and detailed and complicated, she couldn't get the action she needed from the executives. So, what would more often than not happen is they would delete the messages or they wouldn't read them. People don't have a lot of time in the workplace today. Executives especially don't have a lot of time. They're very time-constrained. So, if they see a long email, they may not read it. They just literally don't have the time. So, we work together to develop a methodology whereby she would have the executives take action on the emails and this is the process that we followed. Number 1, I had her clearly state the purpose of the message and put the call to action in the subject line of the email. The next thing that I had her do was identify the three most important points to highlight within the email for communication to the executive and then to include the call to action.

So, what exactly did she want them to do? And then she put all of the detail in the backup. So, she had the detail that she was very sort of wedded to in the email but she had all of the important information upfront. Executives started reading her messages. They started responding. And in this way, she was able to really move the business forward. And that worked very effectively for her. Let's look at the essential aspects necessary for crafting effective professional communication. The purpose of the message is the most essential aspect of professional communication. Your writing must reflect exactly what you want to convey.

To identify the purpose of your message, start with some basic questions. Most importantly, ask yourself what you're trying to achieve with your message. Why are you sending it? Are you informing, persuading, or making a request? Think about what you want the reader to do after they read your message. Based on your answers, your purpose could be to share information, make a request, or persuade the reader to take some action. Sharing information might include explaining how things work, how to get tasks done, or how to solve problems. For instance, memos, user manuals, and product specifications provide directions. This type of writing also provides data to help make decisions like minutes of a meeting, reports, or financial statements. Alternatively, your purpose might be to make a request asking the reader for specific information.

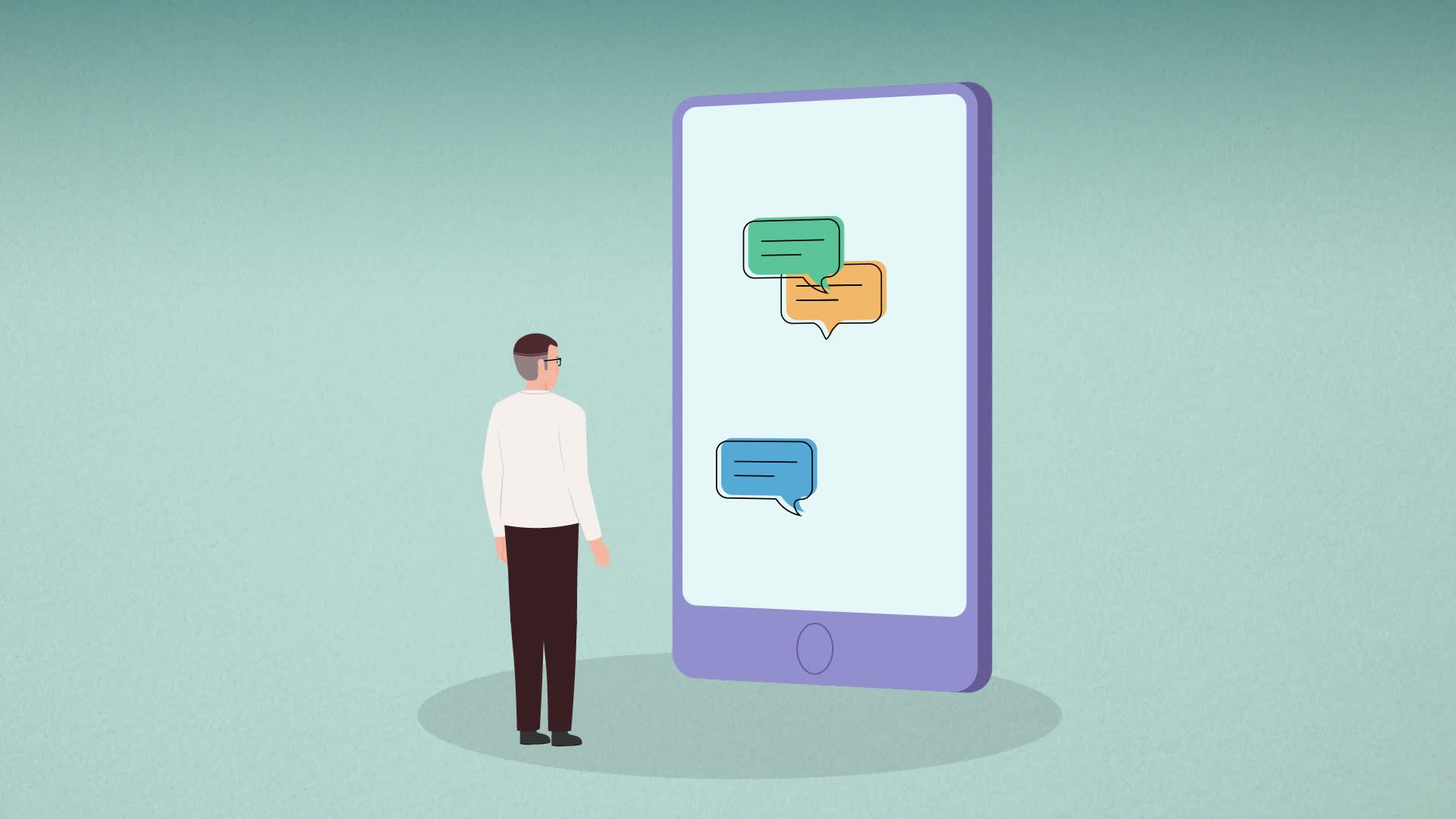
This occurs in daily business communication through emails, letters, direct messages, forms, or invoices seeking responses from colleagues or clients. Another purpose could be persuasion, where you encourage the reader to make a decision or take action, such as buying a product or using a service. This type of writing is aimed at impressing the reader, highlighting the value of a particular product or service. Think about, say, project proposals, sales pitches, marketing emails, advertisements, and press releases. Having a clear purpose guides your language and how you'll structure the message. A single, well-defined purpose enhances clarity and effectiveness, helping you achieve your goals seamlessly. Once you know the purpose of your message, it's important to consider the audience of the message.

In the course of your daily work, you're likely to engage with colleagues from various disciplines. Consider whether you're addressing the message to a vice president or a team member. Understanding your audience will help you gauge the appropriate level of communication and tailor your writing accordingly. Depending on your audience's background, knowledge level, and priorities, you can determine the tone, formality, and content of your written communication.

Apart from the purpose and audience, the tone of the message is another essential aspect of professional communication. Knowing when to use a conversational tone versus a professional tone in your writing is very important. In technical documents, you might dive into complex terms and scientific jargon, but in everyday business communication, it's perfectly fine to use simple words and even contractions. When dealing with investors, a formal tone might be necessary, but when engaging with customers, keeping it conversational often works better.

Always be respectful and polite. Readers will be less receptive if your language doesn't convey respect and professionalism. Use appropriate greetings, titles, and names in formal emails and letters. Be sensitive to cultural differences and wherever it makes sense use gender-neutral language. Be aware of any unconscious biases that might sneak into your writing. Try to focus on the positive rather than the negative whenever possible. Being clear about your purpose and audience and using the right tone can increase efficiency and speed up decision-making. Think about your own communications. Do you consider your audience when writing a professional email? How do you adjust the tone of your message for your audience?

## 3. Video: Crafting Impactful Professional Communication (bs\_acm36\_a01\_enus\_03)

****

If you know why you’re writing and who you’re writing to, you can then move on to how you’ll write. Focusing on your writing is crucial because it’s likely to affect the response you receive. In this video, you'll learn strategies for crafting effective and professional written communication.

* *recognize strategies for crafting effective and professional written communication*

[Video description begins] *Topic title: Crafting Impactful Professional Communication.* [Video description ends]

If you know why you're writing and who you're writing to, you can then move on to how you'll write. Focusing on your writing is crucial because it's likely to affect the response you receive from your colleagues, managers, or customers. To get the desired result from your audience, you need to recognize strategies for crafting effective and professional written communication. Studies indicate that our attention spans have dwindled over the years, now standing at just around nine seconds. That's why it's a good idea to provide the most important information first. Put the key takeaways or summary of your message at the start. Whether you're communicating via email, text, or in a collaboration space, it's best to start with the "why" behind the message.

While most messages are short, some require a messaging hierarchy where you put your most important information first and add supporting details later. State your purpose either in the subject line or the opening line of your communication. Using brief bullet points to support your main message can help decision-makers reach a conclusion quickly. I think a really important starting point for communication is determining the "why". So, what's the purpose of the communication? And then being able to build a case for that. I was working with a senior director at a very large Fortune 100 company, and he had a large team in India, and he was having a very difficult time getting travel approval to go to India to meet with this big team of 300 people.

So, as we talked a little bit more, one of the things I immediately noticed is that he was very happy and he laughed a lot, and he was jovial and uplifting and everybody wanted to talk with him, but he couldn't get any of his initiatives through because nobody really took him seriously. The vehicle for the communication was incredibly important. So, the fact that he was having this conversation with his manager verbally and he was very light-hearted and he laughed a lot, that was part of the problem. And so, we talked a lot about the vehicle that he should use to communicate with his manager. So, he would tell his managers that he needed to go for team building, and they just didn't understand the purpose. They didn't think that was a good enough reason for him to make an expensive trip to India.

So, we worked on really defining the purpose of his trip. And so, he said, well, I need to go because I have to deliver the vision. I have to make sure that all of my individual contributors and my managers are lined up behind the vision, and they have to be able to say no to non-essential activities. And then we built a business case for why that would help to move the business forward and we looked at the increases in efficiency and effectiveness on getting the team on the same page and everybody moving in one direction. So, as a result of that, he was able to get his travel approved and he made the trip to India, and he developed relationships with his management team. And he had identified all of his high-potential performers on the team. In business communication, it's important to be formal and direct. Using the active voice makes the content more direct and easier to understand. Ensure that the information you provide is relevant, value-additive, and complete.

If your content is irrelevant to the audience or contains inaccurate facts, it undermines your message. Avoid slang and other language that might sound informal or offend the reader. Don't use jargon, buzzwords, or acronyms. Your reader may be unfamiliar with these and lose interest in your content. Use "you" to address the reader, "I" to express your personal viewpoint, and "we" to refer to your company. Given the overload of information we all experience, be clear and concise. This helps people to quickly understand and act upon your message. Cut unnecessary words and sentences that don't support your main point. Avoid using too many prepositions. For example, you could use the word viewpoint instead of point of view. Use action words wherever possible, for instance, instead of provided protection to, write protected, before you send, review your message for clarity and concision.

Another key strategy for crafting your professional written communication is to use the appropriate tone. Although professional writing tends to be formal, it's perfectly fine to tone it down for informal messages. But when addressing someone in a position of authority or whom you don't know well, it's best to maintain a formal tone. Once you pick a tone, use it consistently throughout your document or message. Choose words that resonate with your audience and their interests. For example, when drafting a press release or newsletter, you'll likely opt for friendly language. For websites and presentations, use a conversational tone to create a dialogue with the audience. If your organization has a writing style guide, consult and follow it to ensure the right tone and vocabulary.

And to the extent possible, keep the message short. Research indicates that our brain's processing is highest when reading short and simple content. When you keep your sentences short and simple, using familiar words, your readers don't have to exert themselves to understand your meaning. Try to get your point across in as few words as possible. Use short sentences, 25 words or fewer, and paragraphs with no more than 3 or 4 lines. Maintain one point of focus in each paragraph. Consider breaking long sentences into bullet points and using a readability tool to reduce the length and complexity of sentences. Think about business communications you've gotten in the past. How do you react when you get a long, rambling email that takes forever to get to the point? How could you make such an email short and simple? What steps do you take to review and revise your own messages?

## 4. Knowledge Check: Developing Professional Writing Skills

* *discover the key concepts covered in this course*
* *identify the essential aspects necessary for crafting professional communication*
* *recognize strategies for crafting effective and professional written communication*

### **Question 1: Multiple Choice**

**What are the three essential aspects needed for effective professional communication?**

#### **Options:**

1.

The purpose of the message

2.

The audience for the message

3.

The tone of the message

4.

The device used to write the message

5.

The inclusion of personal anecdotes

#### **Answer**

1.

The purpose of the message

2.

The audience for the message

3.

The tone of the message

#### **Feedback:**

*Option 1:*

*This option is correct. The purpose of the message is an essential aspect because it defines the goal and direction of the communication. Your writing must reflect exactly what you want to convey. For example, your purpose may be to share information, make a request, or persuade the reader to take action.*

*Option 2:*

*This option is correct. The audience for the message is an essential aspect necessary for crafting professional communication because it influences the tone, content, and relevance of the message. Understanding your audience will help you determine the tone, formality, and content of your communication.*

*Option 3:*

*This option is correct. The tone of the message is essential for crafting professional communication because it influences how the message is received and interpreted by the recipient. Consider whether the tone should be conversational or professional, use appropriate greetings and titles, and always be respectful and polite.*

*Option 4:*

*This option is incorrect. The device used to write the message is not an essential aspect of effective professional communication because the choice of device does not affect the content of the message.*

*Option 5:*

*This option is incorrect. The inclusion of personal anecdotes in a message is not an essential aspect of effective professional communication. Professional communication is about communicating business information, and is different from informal personal conversations.*

### **Question 2: Multiple Choice**

**What are five useful strategies for crafting effective and professional written communication?**

#### **Options:**

1.

Provide the most important information first

2.

Be formal and direct

3.

Be clear and concise

4.

Use the appropriate tone

5.

Keep the message short

6.

Engage in a lengthy narrative

7.

Save the most important information for the end

#### **Answer**

1.

Provide the most important information first

2.

Be formal and direct

3.

Be clear and concise

4.

Use the appropriate tone

5.

Keep the message short

#### **Feedback:**

*Option 1:*

*This option is correct. Providing the most important information first is a strategy for crafting effective and professional written communication because it immediately grabs the recipient's attention and ensures clarity of the main message.*

*Option 2:*

*This option is correct. Being formal and direct conveys professionalism, clarity, and respect for the recipient's time and expectations. Use the active voice to make content more direct and avoid slang, jargon, buzzwords, and acronyms.*

*Option 3:*

*This option is correct. Being clear and concise ensures that the message is easily understood and minimizes the risk of misinterpretation or confusion. Cut unnecessary words and sentences and use action verbs whenever possible – for example, saying "protected" instead of "provided protection to."*

*Option 4:*

*This option is correct. Using the appropriate tone is important to effective and professional written communication because it conveys the desired attitude, emotion, and level of formality while aligning the message with its intended purpose and audience.*

*Option 5:*

*This option is correct. Keeping the message short is a strategy for crafting effective and professionally written communication because short messages ensure that key information is easily comprehended.*

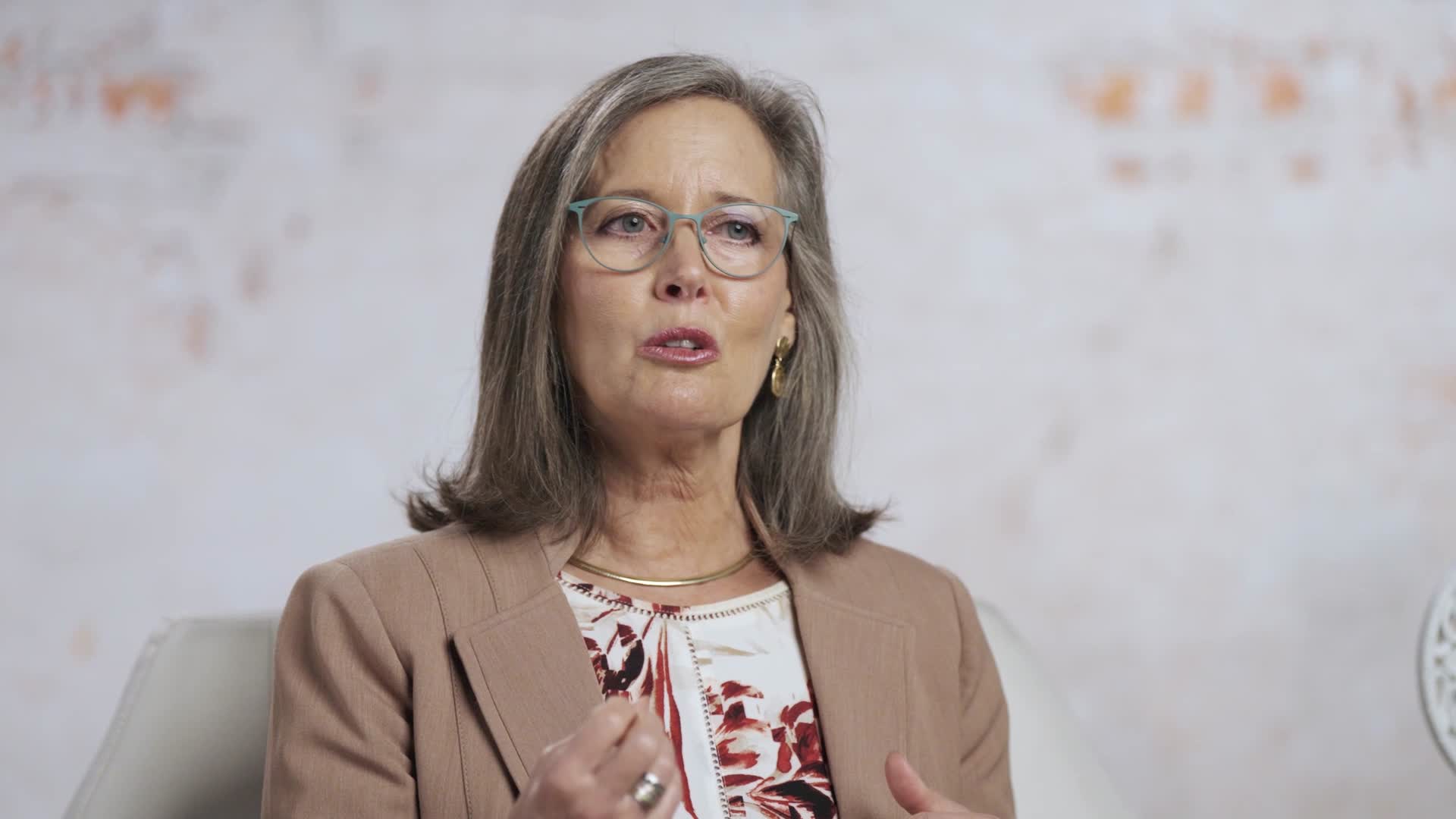
*Option 6:*

*This option is incorrect. Engaging in a long narrative is not a strategy for crafting effective and professional written communication. When content is unnecessarily lengthy, the reader may lose interest and miss the key message.*

*Option 7:*

*This option is incorrect. Providing the most important information last is not a strategy for effective and professional written communication because the reader may miss key information if it is presented at the end of the message.*

## 5. Video: Tailoring Your Communication for Different Channels (bs\_acm36\_a01\_enus\_04)

****

Most professionals use different communication channels for internal and external communication. It’s important to adapt your writing based on the communication channel you're using. In this video, you'll learn how various channels of communication can be optimized for impactful professional writing.

* *recognize how various channels of communication can be optimized for impactful professional writing*

[Video description begins] *Topic title: Tailoring Your Communication for Different Channels.* [Video description ends]

So, the communication, the form of the communication, the form of the written communication, as well as the channel is critically important when you're working on large, complex initiatives. You typically have multiple audiences who need to receive the message. You have different requests or asks for each audience. And so, what we really did as a team, we were getting ready to launch a major initiative to deliver a $100 million to the company within 12 months. And so, we had to be crystal clear on our communications. And so, the first thing that we did was we identified all of the audiences who would need to receive the communications, and then we determined what channel we should use to reach those executives, and then what the call to action was for each audience and then what our expected outcomes were.

So, for example, when we had large executive readouts, we made sure that we were sending them an email and a PowerPoint presentation. So, a clear written communication in advance of the executive readout, and we would hold the meeting, have the presentation, they could see it in advance, and then we followed up via email with action items and next steps. For urgent requests for executives, we would always use email, but we would also follow up with a voicemail and a text, and then we would contact their executive assistant if necessary, to schedule a meeting. For large team readouts, we would put those messages in collaboration spaces, which are broadly used, of course, today.

But in those collaboration spaces, it was really clear that we outlined the purpose of the communication and then what the calls to action were and the roles and responsibilities so who the owners were. Within broad-based communications to the entire company, we would again send an email with embedded links, and then we would also include videos to make sure that the messages were very clear, and they had multiple ways to consume the information, and then we would have calls to action. Externally when we were communicating with customers, with partners, with the broader market, we would either communicate through the salesforce or our partners, again, giving them the appropriate information in writing, and then we would brief the press. So, making sure that we were very clear again what the purpose of the communication would be or what it was, and the calls to action.

And so, that helped us to be very effective as a part of this initiative. And we ended up delivering a $184 million year over year. So, it was very successful. Communications, written communications was a critical component of that. Most professionals use different communication channels for internal and external communication. The communication channel you choose may depend on the culture of your organization. While some organizations use emails extensively, other organizations mandate all communication to go through collaboration spaces only. In any case, it's important to adapt your writing based on the communication channel you use. To do this, you need to recognize how various channels of communication can be optimized for impactful professional writing.

Email tends to be the most formal form of written communication to correspond with specific recipients and create a record of communication. Crafting effective emails is essential to ensuring a smooth flow of important information. A well-crafted email helps the sender showcase their professionalism and strong communication skills while sharing their key message. An email reading, I need your feedback on the draft agenda. Please send your edits by 10 a.m. tomorrow, states the message and expectations clearly. Avoid the common mistake of providing too much information. Don't make your audience search for your request. Consider what matters most to your reader and highlight it. Put the main point or ask in the first few sentences and then provide the details and context. You can call out the ask in the subject line to let the recipient know that they're expected to respond to the email.

Another common channel is using text messages, team chats, or collaboration spaces. These serve as platforms for quick, informal updates. They're mostly used for sharing project progress within a team. In large collaboration spaces with many members, however, it's common to encounter issues. There's often confusion about who should act on a request and its priority. Requesters might feel frustrated when their messages go unanswered in these types of forums, where it's easy to miss information. To avoid this situation, be open and transparent about what you're asking for and clear about whom specifically you're addressing. And remember, these are often seen by a lot of people. Temper your language and stay positive. Avoid being negative in a team chat.

If you need to be harsh to be direct, do it through a private one-to-one channel. In contrast presentations are commonly used to communicate with a large audience. They're used to outline and advance program initiatives or tell a story to higher-ups or customers. To create an effective presentation, develop a goal and key takeaways based on what you want your audience to learn, feel, or do after the presentation. Structure your presentation around the central theme. You can start with an attention-grabbing introduction to the main point, explain why it is important, and list the key takeaways. Use stories, examples, and evidence to explain each main point. End the presentation with a strong summary, restating the ask with a call to action. Use storytelling techniques like setting a scene, showing a conflict and resolving it, and sharing lessons learned to engage your audience.

Finally, website content and blogs are communication channels that help communicate the value you provide to your customers and shareholders. They are best suited for sharing information with an external audience. It can help to use survey or marketing data to create customer personas that represent your target audience. Then you can adjust your tone and language to suit that audience and what they want. Here again, you can appeal to the emotions of the reader by structuring your content around a story.

You want to find ways of keeping your reader hooked. Make them want to keep reading to find out what happens next and how your products or services can help them solve their problems. Different communication channels meet different communication needs, whether it's getting information for tomorrow's finance meeting or convincing customers your shoes are the most comfortable. You can optimize your message by choosing the right channel and adjusting your writing to best take advantage. Think about your job. Which communication channels do you use most often? How does your audience respond to your messages? How can you adjust your language and tone to best suit the channel?

## 6. Video: Leveraging Best Practices to Improve Writing Skills (bs\_acm36\_a01\_enus\_05)

****

Whether we like it or not, people judge us based on how we communicate – our tone, style, level of detail, clarity, and relevance to the business. We’re also judged on accuracy and grammatical errors in our messages. In this video, you'll explore best practices for leveling up your professional writing skills.

* *recognize best practices for leveling up your professional writing skills*

[Video description begins] *Topic title: Leveraging Best Practices to Improve Writing Skills.* [Video description ends]

Whether we like it or not, people judge us based on how we communicate our tone, style, level of detail, clarity, and relevance to the business. We're also judged on accuracy and grammatical errors in our messages. It's important to think about how to ensure clarity, grab the reader's attention, and project positivity and confidence in our writing. And that's why you need to recognize best practices for leveling up your professional writing skills. Media training is one of the most valuable experiences that I've ever been through in my entire life. If you ever have a chance to go through media training, I would highly recommend it because it absolutely feeds everything that you do from a written communication's perspective, and it's pretty simple and straightforward.

And the tool for media training is the inverted pyramid. And so, what that means is that you put your most important information at the top of the pyramid. So, let's think of an example. For example, you're working on a disaster preparedness program. And you know that if people implement this program and what the requirements are of the program, you're going to increase the number of lives saved by 75%. So, that becomes the top of your pyramid, that implementing this program will result in a 75% increase in life saved if we have a disaster. I think one of the most important reasons to have a simplified message and to think in terms of this inverted pyramid is because you won't be searching around for extraneous detail, and it's much easier to remember.

So, you can repeat the message to anyone. Number 1, by implementing this plan, you can increase life savings by 75%. You need to create a plan. You have to be able to determine the immediate actions. And then you have to have a communications plan to follow up. So, when you can do this, you come across in a much more positive and confident way than if you didn't have a simple process and methodology to communicate. There's a common saying that great writing is rewriting. While you may not always have time to rewrite on the job, it's best practice to organize your thoughts before you start writing. Avoid organizing while you're writing. That tends to make your writing less structure, repetitive, and more likely to drift off track.

A good way to start is to brainstorm about everything you think is important to include. You could do this on a piece of paper or another document. List your ideas until you identify your key point and information. This becomes your focus. Everything else you include should support that key point and help your readers understanding. Then outline your content and start writing. Most people want you to get right to the point, you need to grab the reader's attention quickly. This means your content must be relevant, well-structured, and concise, keeping your writing free of factual and grammatical errors and providing only relevant data establishes your credibility. Using short, punchy sentences, and simple syntax helps keep your audience reading. Positive messages and phrases appeal to readers.

It's best practice to project positivity and confidence in your writing. When you use positive phrasing, your readers are more likely to respond positively. Positive messaging promotes collaboration and willingness to listen to your ideas. When people are approached with positivity, it changes the way they react. Avoid using an accusatory tone. Don't assign blame for problems, but rather focus on ways to mitigate and fix them. Reframe the negative as opportunity. Focus not on this thing went wrong, but instead on here's a chance to improve this thing. Finally, take the time and effort to proofread and edit carefully. Grammatical and spelling errors put your message in a poor light.

Good writing presents you as professional and competent. Proofreading and editing both improve your writing and let your audience focus on what you have to say without being distracted by errors. Here's a secret no one notices good writing, but everyone notices bad writing. Use spelling and grammar checks if you have them. Make sure everything makes sense and says what you want it to say. If you have time and it's an important piece of writing, for example, for press release or in a presentation, get someone else to have a look. A fresh set of eyes can find things you may miss. Stay far away from awkward fonts and formats, including those copied and pasted from other documents. Avoid using underlining in emails or web content. Readers can confuse the text with hyperlinks.

For lengthy documents and websites, consider using screen readers to ensure that your content meets accessibility standards. In many ways, good professional writing is just good writing period. But the needs of the workplace aren't exactly the same as the needs of a good novel or an engrossing magazine article. Professional writing must meet the needs of the professional doing the writing and their audience, but with the right mindset and by following a few best practices, you can up your business communications and come across as the pro you are. Now consider for a moment your own professional writing habits. How often do you ask your colleagues' opinions on your content? Do you ask for proofreading help? What's your process before hitting the send button?

## 7. Knowledge Check: Enhancing Professional Writing Skills

* *recognize how various channels of communication can be optimized for impactful professional writing*
* *recognize best practices for leveling up your professional writing skills*

### **Question 1: Matching**

**Match each communication channel to its best method of optimization.**

#### **Options:**

A.

Email

B.

Text messages, team chats, and collaboration spaces

C.

Websites and blogs

D.

Presentations

#### **Targets:**

1.

Specify the ask in the first few sentences and call it out in the subject line

2.

Be clear about the ask and who should act

3.

Develop the main theme and structure around telling a story

4.

Evaluate the audience and structure around theme and primary ask

#### **Answer**

1:

Option A

2:

Option B

3:

Option C

4:

Option D

#### **Feedback:**

*Target 1:*

*Specifying the ask in the first few sentences and calling out the ask in the subject line is the best way to optimize email. It ensures that messages are well-received and chances of miscommunication are reduced.*

*Target 2:*

*Being clear about the ask and who should act or respond is the best strategy for text messages and collaboration spaces. It helps to ensure effective communication in these potentially broad-reach platforms while aligning with the organization's values and culture.*

*Target 3:*

*Developing the main theme or refrain and structuring around telling a story is a good way to best optimize website content and blogs. It engages readers, aligns with marketing and PR goals, and effectively conveys the desired message to the target audience.*

*Target 4:*

*Evaluating the audience and structuring around a theme and the primary ask is the best way of optimizing presentations. It ensures that the content is tailored to the audience's needs and effectively conveys key information.*

### **Question 2: Multiple Choice**

**What are four best practices for leveling up your professional writing skills?**

#### **Options:**

1.

Organize your thoughts before writing

2.

Grab the reader’s attention

3.

Project positivity and confidence

4.

Proofread and edit carefully

5.

Write your thoughts out as you go

6.

Make spelling and grammar a secondary concern

#### **Answer**

1.

Organize your thoughts before writing

2.

Grab the reader’s attention

3.

Project positivity and confidence

4.

Proofread and edit carefully

#### **Feedback:**

*Option 1:*

*This option is correct. Organizing your thoughts before writing ensures that your writing is well-structured, concise, and focused, avoiding the pitfalls of unstructured, repetitive, and off-topic content that can detract from the effectiveness of your message.*

*Option 2:*

*This option is correct. Grabbing the reader’s attention is a best practice for leveling up your professional writing skills because it helps avoid verbosity and ensures that your writing conveys relevant and useful information to the target audience, making your message more engaging and effective.*

*Option 3:*

*This option is correct. Projecting positivity and confidence helps you consider the impression recipients might form about you from your messages. A positive tone appeals to readers, fostering a more favorable and effective communication experience.*

*Option 4:*

*This option is correct. Proofreading and editing carefully is a best practice for leveling up your professional writing skills. Content with grammatical and spelling errors reflects poorly on your abilities and can distract the reader from your message, compromising the overall impact of your communication.*

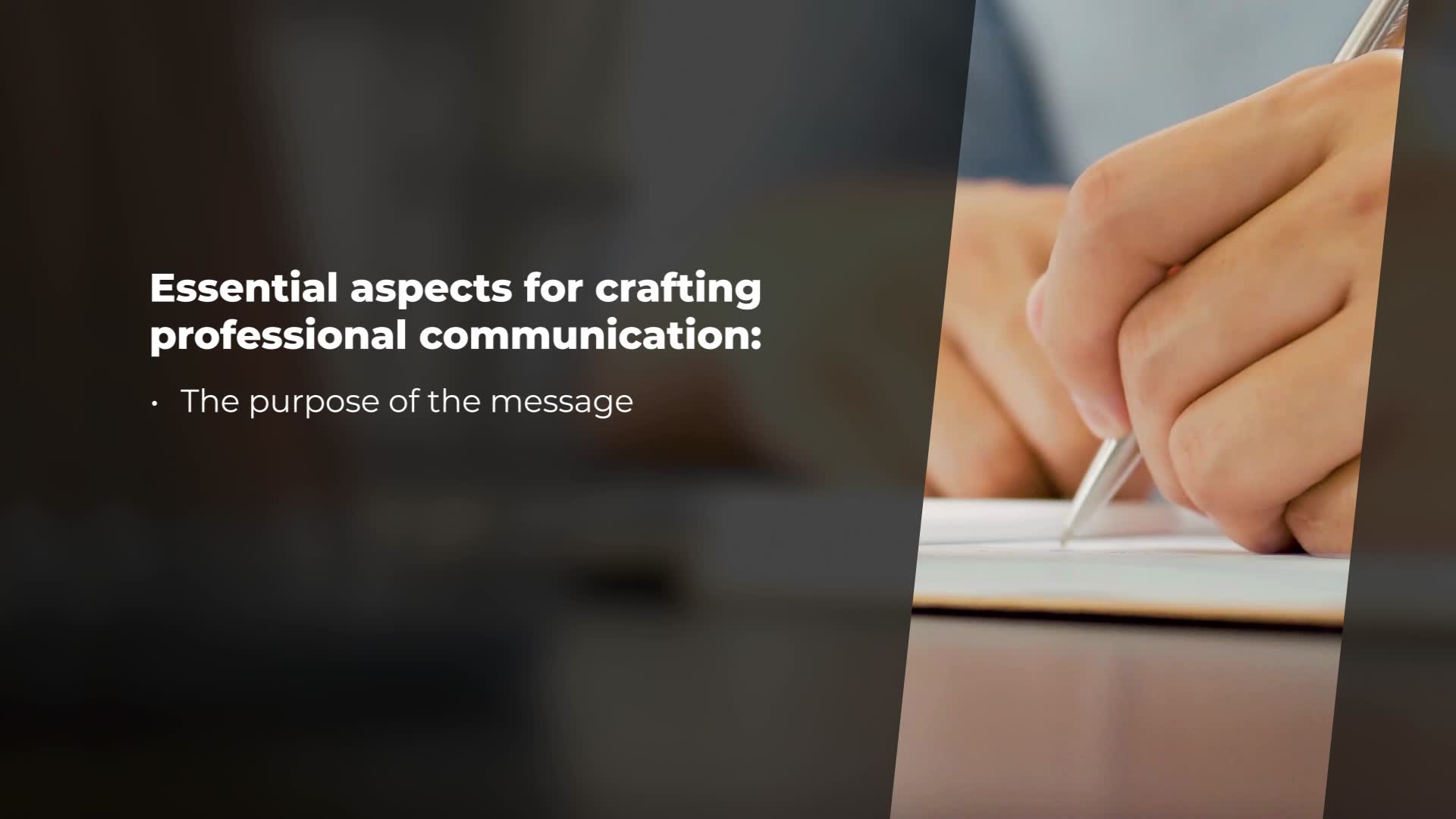
*Option 5:*

*This option is incorrect. Thinking only while writing is not a best practice to level up your professional writing skills because when you think while writing, the content could become less structured and more repetitive, and may meander off-topic.*

*Option 6:*

*This option is incorrect. Ignoring or minimizing spelling and grammar is not a best practice to level up your professional writing skills. If the content has errors, it might distract the reader and create a bad impression of the author.*

## 8. Video: Let's Review (bs\_acm36\_a01\_enus\_06)

****

In this video, you'll review and reflect on what you've learned in the course Writing with a Professional Mindset.

* *reflect on what you've learned*

[Video description begins] *Topic title: Let's Review.* [Video description ends]

Let's review what you've learned in this course. The essential aspects necessary for crafting professional communication are the purpose of the message, the audience for the message, and the tone of the message. Strategies for crafting effective and professional written communication include providing the most important information first, being formal and direct by using active voice and avoiding slang and jargon, being clear and concise, using the appropriate tone, and keeping your message short.

You learned how various communication channels can be optimized for impactful professional writing. These include using email for formal communication and using text messages, team chats, and collaboration spaces for quick updates. There are also presentations for large programs and initiatives, and website content and blogs to communicate the value you provide to customers and shareholders. Finally, there are some best practices for improving your professional writing skills. They include organizing your thoughts before writing, grabbing the reader's attention, projecting positivity and confidence, and proofreading and editing your content carefully.

## Course HTML Resources

| • | Glossary: Writing with a Professional Mindset |
| --- | --- |
|  |  |

| **accessibility standards** | The regulations and guidelines that ensure equitable access to digital content and physical spaces for individuals with disabilities. |
| --- | --- |
| **brainstorming** | The process of generating creative ideas, solutions, or suggestions by a group of individuals through open and spontaneous discussion and idea sharing. |
| **clarity** | The quality of communication that ensures unambiguous and straightforward information or messages. |
| **collaboration space** | A physical or digital environment designed to facilitate teamwork and cooperation among individuals or groups through tools and resources for sharing ideas and working together on projects. |
| **communication channel** | A medium or method through which information is transmitted from a sender to a receiver, such as email, telephone, or face-to-face conversation. |
| **concision** | The practice of conveying information or ideas clearly and succinctly, using as few words as necessary to communicate effectively. |
| **jargon** | The specialized terminology, vocabulary, or language used within a particular field, profession, or social group that hinders understanding for those unfamiliar with it. |
| **professional writing** | The creation of written materials that adhere to established standards, conventions, and context-specific expectations, ensuring clarity, accuracy, and appropriateness. |
| **proofreading** | The act of carefully reviewing and correcting written content to eliminate errors in spelling, grammar, punctuation, and formatting, ensuring the document is polished and error-free. |
| **readability tool** | A software or tool designed to analyze and assess the clarity, comprehension, and ease of reading written text, often providing suggestions for improvement. |
| **screen reader** | An assistive technology software or device that converts digital text and content into audible speech or Braille, enabling individuals with visual impairments to access and navigate digital content. |
| **slang** | Informal and unconventional language or expressions typically used within a specific social group or subculture, often evolving rapidly and not found in formal language. |
| **tone** | The emotion or attitude conveyed through written or spoken language to influence the perception and impact of a message. |

© 2025 Skillsoft Ireland Limited - All rights reserved.

|  |  |
| --- | --- |

Effective email communication is crucial in the workplace and can save time, prevent misunderstandings, and enhance your professional image. Use these practical tips to improve your email etiquette and make your messages clearer and more efficient.

| **Tip** | **Actions** |
| --- | --- |
| Keep emails relevant and concise | * Keep emails concise and provide only essential details * Ensure that every email, if possible, covers only one topic or request * Use bulleted or numbered lists to cover more than one topic or request * Consider what matters most to your reader and highlight that topic |
| Start strong | * Begin your email with the main point or request * Follow with additional details and context * Avoid burying important information at the end |
| Use the subject line effectively | * Use a clear and descriptive subject line * Include a call to action or the main request * Specify the request or topic to gain the recipient's interest * Ensure that a single email thread is used for the same topic |
| Summarize complex emails | * If you receive a lengthy or disorganized email, summarize the main points in your reply * Help the sender by organizing their thoughts |
| Hyperlink effectively | * Consider using hyperlinking (Command K on Mac or Control K on Windows) when sharing links; this creates a cleaner look and reduces the chance of errors in long URLs |
| Adjust default email settings | * Change the default setting to "Reply" instead of "Reply all" to minimize the risk of accidental emails to all recipients |
| Adjust Undo Send option | * Extend the "Undo Send" option to allow for corrections after sending * Adjust the "Undo Send" option to a longer timeframe (e.g., 30 seconds), to identify mistakes |

Additional tips:

* Remember that email messages are often formal and create a record of communications
* Evaluate your audience and understand their priorities
* Be clear and brief in your communication
* Spell out acronyms and avoid excessive abbreviations

[Open in new tab](https://codeskillpes.percipio.com/courses/8355df47-0315-4045-b673-6929dc9bfb90/videos/c1e2ccc4-8c68-40d5-950f-fc99fb82c55c)

**27 - Telling a Business Story**

# Telling a Business Story

As humans, we've been using narratives to share our knowledge and experiences with others since the beginning. But effective storytelling, specifically within a business context, requires thoughtful preparation and effective delivery. With practice, using storytelling can improve the way you communicate with key players inside – and outside – your organization.  
In this course, you'll learn how to plan and deliver effective stories that engage your target audience and help you accomplish your objectives.

## Table of Contents

[1. Video: Telling a Business Story (bs\_acm21\_a01\_enus\_01)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_0)

[2. Video: Why We Tell Stories (bs\_acm21\_a01\_enus\_02)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_1)

[3. Video: Elements of Effective Stories (bs\_acm21\_a01\_enus\_03)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_2)

[4. Video: Know Your Audience (bs\_acm21\_a01\_enus\_04)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_3)

[5. Video: Plotting Your Story (bs\_acm21\_a01\_enus\_05)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_4)

[6. Video: Structuring Your Story (bs\_acm21\_a01\_enus\_06)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_5)

[7. Video: Telling Your Story (bs\_acm21\_a01\_enus\_07)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_6)

[8. Video: Framing Your Story (bs\_acm21\_a01\_enus\_08)](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_7)

[9. Knowledge Check: Telling Business Stories](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_8)

[Course HTML Resources](https://cdn2.percipio.com/secure/c/1764060695.f2598f9041bd7e96b5fc47b7653f968bfad08053/eot/transcripts/ea978eb4-c1e8-4a1a-a935-b476f2fdf305/bs_acm21_a01_enus.html#section_10)

## 1. Video: Telling a Business Story (bs\_acm21\_a01\_enus\_01)

****

After completing this video, you will be able to discover the subject areas that will be covered in this course.

* *discover the subject areas that will be covered in this course*

[Video description begins] *Topic title: Telling a Business Story.* [Video description ends]

Never underestimate the importance and impact of a good story. As humans, we've been using narratives to share our knowledge and experiences with others since the beginning of time. But effective storytelling, specifically within a business context, requires thoughtful preparation and effective delivery. With practice, using storytelling can improve the way you communicate with those inside – and outside – your organization.

In this course, you'll learn the benefits of using stories to present business information and the structures and strategies for drafting and writing your business story. You'll also explore how to plan and deliver effective stories that engage your target audience and help you accomplish your objectives.

## 2. Video: Why We Tell Stories (bs\_acm21\_a01\_enus\_02)

****

After completing this video, you will be able to recognize the benefits of transmitting information via storytelling

* *recognize the benefits of transmitting information via storytelling*

[Video description begins] *Topic title: Why We Tell Stories.* [Video description ends]

Whether you're selling a product online or explaining a concept to colleagues, one of the best ways to get your message across is through a story. Storytelling is second nature to us. We tell stories – and listen to them – every day. It's how we learn, teach, communicate, entertain, and understand. And in business too, there are real benefits to using storytelling to communicate information.

Why is storytelling such an effective way of transmitting information? One reason is that storytelling engages every part of the human brain. Unlike a non-narrative presentation, a well-told story speaks to both the left and right sides of the brain. Hence, the listener can process an event and analyze it at the same time. It's because storytelling is such a natural human process.

Another reason is that storytelling fosters deeper connections between people, which ultimately builds empathetic relationships with the audience. In contrast to a standard, formulaic presentation of information, an engaging story fosters a sense of camaraderie and shared experience.

For example, you've probably seen how sharing stories between team members is a great way to build team spirit and empathy. Suppose with a team project well-underway, you discover that the requirements are suddenly changing, negating several months' worth of work.

Sharing a story of a previous project in which a similar abrupt change of direction ultimately led to a greater level of project and company success, can help foster empathic connections and facilitate understanding and mutual support, which in turn mitigates against the potential for frustration and anger.

And that brings us to the next reason why storytelling is so effective at communicating information: it engages the listener on multiple levels. Notice how engaged listeners lean forward when listening to a story. That's because a good story engages us fully: physically, mentally, emotionally, and spiritually. Appealing to a listener's emotions makes any story more memorable.

And finally, storytelling is an effective way of transmitting information because it increases the perceived value of the subject. Storytelling can be particularly useful in sales, for instance. Telling an interesting story about a product makes customers more inclined to buy it. An engaging narrative is often one of the deciding factors when people buy a product or service.

For example, look at how life insurance is marketed: a story or narrative is presented about how insurance protects families and loved ones. A story about something we care about is used to get us to think about a product we'd really rather not think about. And then we buy it. Ultimately, when you connect a business objective to a story, the perceived value and appeal of the product increases.

Storytelling isn't limited to generating sales, of course. It can be used in project management, training, customer service, planning, and virtually any other area. It's a valuable tool for most any business practice. Stories connect us.

It's not as simple as "once upon a time." Using effective storytelling within a business context takes some preparation and practice. It takes effort. But it's worth it. Instead of merely relaying facts, telling a story makes things memorable, understandable, compelling, and relatable. And more than anything, it engages the listener.

## 3. Video: Elements of Effective Stories (bs\_acm21\_a01\_enus\_03)

****

After completing this video, you will be able to identify the elements of an effective business storytelling effort

* *identify the elements of an effective business storytelling effort*

[Video description begins] *Topic title: Elements of Effective Stories.* [Video description ends]

People are exposed to stories all the time. They're everywhere – and even more so since the advent of the Internet and social media. To be heard above the multitude, you need to make your stories stand out from the crowd. And you can do that by ensuring your stories have all the necessary elements of effective business storytelling.

Specifically, there are four elements that work in concert to make a business story effective. The first element is relatability. People are more comfortable with things that they know – things they can relate to. It follows then that your audience will be more engaged with a story that features familiar things – a setting or a character they know and can identify with. If your content requires presenting new ideas or context, find familiar analogies to use or ways to compare it to common examples. Relatability is the secret to truly engaging audiences.

Seemingly conversely, the next element is novelty. While familiarity is good up to a point, and audiences love characters they recognize, repeating the same narrative or storyline over and over gets old and tends to make an audience tune out. New stories, or novel variations on a familiar theme, are more engaging for an audience than ones they've already heard. A new story is naturally more interesting. It refocuses attention and draws the audience back in.

Another element of effective storytelling is tension. A sense of tension in your story will draw your audience in – you can see it in their physical reaction when they lean forward, concentrating, focused and engaged, anxious to find out how the problem will be resolved. They want to know, need to know, what's going to happen next.

But it's a balancing act. You want enough tension to attract and draw in your audience, but not too much tension. Too much tension pushes people away. It takes practice to get it right. The final element is ease of understanding. It's important to keep the language and plot simple so that the story is ultimately easy to understand. The audience won't be engaged if it's struggling to understand exactly what's at stake or what's going on.

Don't put barriers in the way of the audience getting involved and engrossed. Make it easy for them. And when necessary, tailor the story to your audience. Experts and novices are very different audiences, and don't forget that experts in topic A may be novices in topic B.

Storytelling is a powerful, compelling, and engaging way of getting your message across and connecting with your audience. And your business storytelling effort needs to hit the mark. And that means storytelling that has relatability, novelty, tension, and ease of understanding.

## 4. Video: Know Your Audience (bs\_acm21\_a01\_enus\_04)

****

After completing this video, you will be able to recognize actions that create a connection with your audience, increasing the impact of your story

* *recognize actions that create a connection with your audience, increasing the impact of your story*

[Video description begins] *Topic title: Know Your Audience.* [Video description ends]

Every story needs an audience – there's no point to a great story if no one ever hears it. To maximize the impact of your story, you need to identify the right audience and then create a connection with them. While this is easier said than done, there are four specific actions you can take to create a connection with your audience.

In a business context, your audience will generally be defined for you, based on the business need that's being met by your story. They might be stakeholders whose approval you need to jump-start a project. They might be clients you need to persuade to take action on an initiative. They might be coworkers you need to convince of the need for a new business process.

Whatever the case, the first action is to profile the target audience. You need to know and really understand your target audience. Marketers might ask, what's the company's ideal customer? What do they like and dislike? What are their habits? Opinions? Priorities? When and how do they get their information?

Conversely, the audience might be a new cross-functional project team filled with specialists from many functional areas. Or maybe a group of experts in one particular field. In all cases, you need to know who they are and what they want and need to know. Let these details guide and direct your storytelling because these people are your target audience.

Once you've done this, you can then begin to match your message to the target audience. Don't try to create a generic story that will appeal to everyone. It's impossible. In fact, do the opposite. Tailor the story to appeal directly to your target audience. Match the story to their preferences, habits, needs, and prior knowledge.

A group of web developers don't need to be told how a multi-page responsive website works, but the sales team might. Your audience is all that matters. This includes internal, external, and business-to-business audiences. Whoever they are, that's who they are.

The next action is to create details and a narrative to use stories to connect more deeply with that audience. Stories have the power to create real, genuine connections between the storyteller and the audience in a way that statistics, straight facts, or product features struggle to create. Stories are simply more engaging. And that's what your audience wants: to be engaged.

Draw your audience in with a story – don't push it away with a simple, dry recitation of facts and statistics. Engage them in your story, putting a face and narrative to those facts and stats and you build a connection deeper and more meaningful than a list of product features or attributes could ever hope to achieve.

The final action is to use the audience's own words. How the story is told can be as important as the story itself. That means telling the story in a way your audience understands and can relate to – or telling it in their own words, using their own language, terminology, and vocabulary. Using their words will draw them in and connect them to you.

This often means, of course, having different versions of the same story for different audiences. If you're telling the story to a group of accountants, it's okay to use their financial jargon – but if you're telling the story to a tech audience, leave out the debt-service coverage ratio, but it's fine to drop in a DHCP. Ultimately, for your business story to resonate and have a genuine impact, you need to first know and understand your audience. Then you can dovetail your message, your story, with their needs.

## 5. Video: Plotting Your Story (bs\_acm21\_a01\_enus\_05)

****

After completing this video, you will be able to recognize key strategies for the initial stages of planning a story

* *recognize key strategies for the initial stages of planning a story*

[Video description begins] *Topic title: Plotting Your Story.* [Video description ends]

Every story needs a plot. You can't just jump in without a clear goal, interesting ideas, believable characters, and a moral. And that's why it's so important to plan your story. And the initial planning stages are especially important.

Fortunately, there are some key strategies for the initial stages of planning a story. The first key strategy is to set goals for the story. What's the point of your storytelling effort? What are you trying to achieve? You need to understand what gap your story will fill, what connection it will make. Why are you telling the story in the first place? Are you trying to sell something? Are you trying to sway someone's opinion? What's the point?

Once you've figured this out, you need to decide how you can best tell your story. Should you go for humor or pathos? Do you want your audience to laugh or cry? And what's your format? Are you talking in front of a whiteboard or shooting a video? These matter in making your point.

Next, brainstorm ideas. Perhaps you already have some rough or vague ideas, but outside input helps. Brainstorming can bring out aspects of a story that you have overlooked or missed. It's better to have lots of ideas than just a few – you can whittle things down later. Be sure, though, not to lose focus – so keep the goal and the audience in mind.

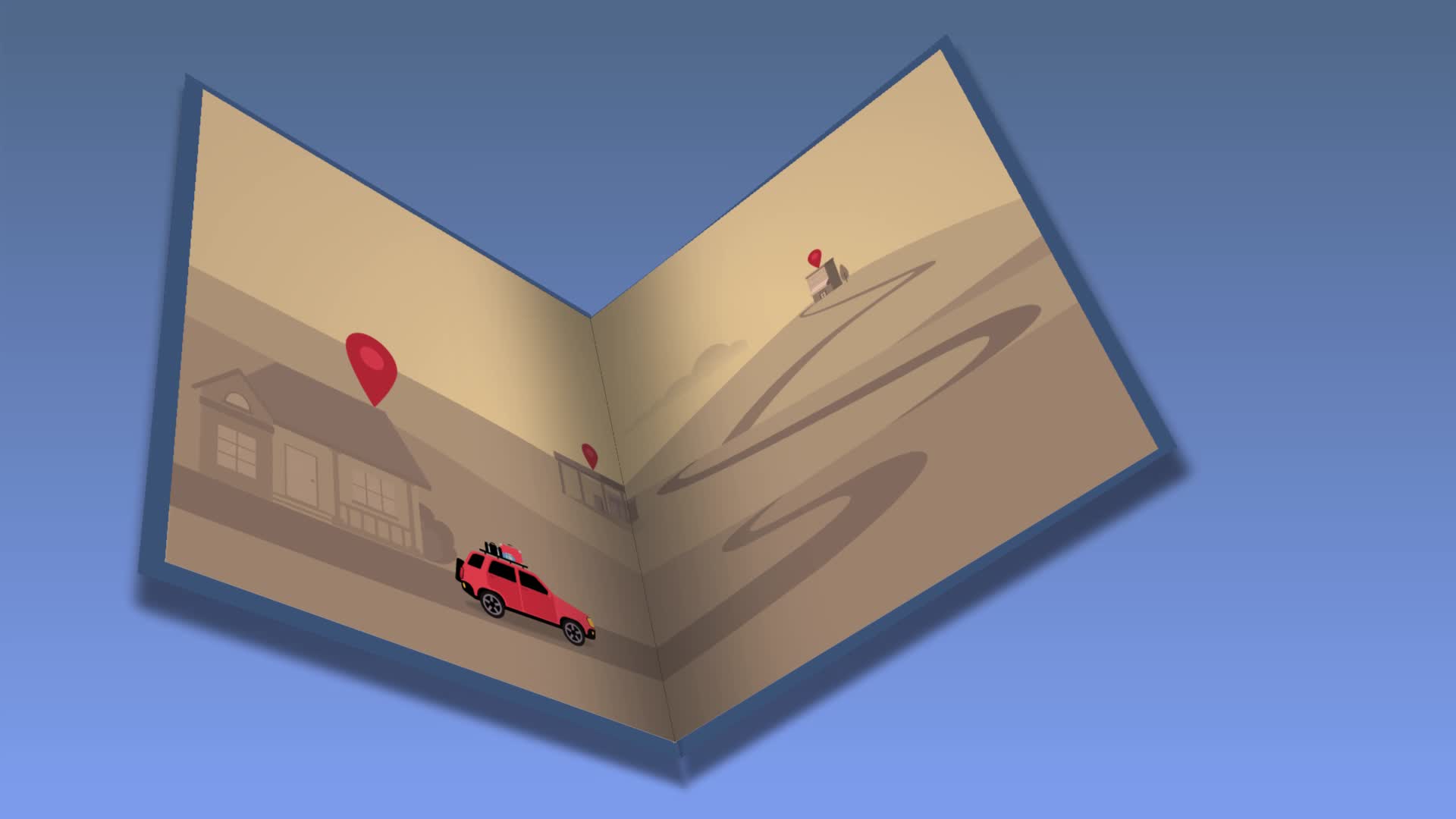
Once you've decided on your idea, iron out the details. Have a clear story pitch. Outline the hero, the plot, and moral of the story. It's your story – sell it. The next strategy is to create characters. Every story needs a hero – a main character. The hero – or main character – delivers the message of the story.

Your story can have as many characters as necessary, but don't go overboard. Get rid of any unnecessary characters, and make sure they're distinct and that the audience understands what makes them tick. And most of all, keep them believable. Never forget that your hero represents your message, your company, your brand, your product. The hero doesn't have to wear a logo or anything too obvious, but do ensure that the hero represents or embodies the qualities of the brand or product.

The final strategy is to identify the moral. It's just that: what's the moral of the story? What do you want the audience to take from the story? That's the value judgment about your brand or product you want the audience to make. There's a balancing act here: on the one hand, you want the moral to be clear enough that everyone gets it, but subtle enough that the audience doesn't feel judged or talked down to. They should be able to discern the moral on their own.

One last thing to note: when you're figuring out the moral of the story, keep the audience in mind. Make sure it's compatible with their feelings of right and wrong, for instance. In every aspect of business, planning is crucial. And effective business storytelling is no exception. Before you take your story anywhere, you need to know where you want it to go.

## 6. Video: Structuring Your Story (bs\_acm21\_a01\_enus\_06)

****

After completing this video, you will be able to identify strategies for outlining the beginning, middle, and end of a story

* *identify strategies for outlining the beginning, middle, and end of a story*

[Video description begins] *Topic title: Structuring Your Story.* [Video description ends]

A story that lacks direction or shape, one that fizzles out before the end, is a story that's not going to serve its business purpose. That's why it's so important to properly structure your story – to take it from point A to point Z smoothly and effectively. This means clearly outlining your story's beginning, the middle, and the end. And for that, there are some useful strategies.

When looking to effectively structure your story, the first strategy is to determine the key message. Essentially, it's the fundamental point of the story. And it comes out of the theme. For example, if the theme is fiscal responsibility, the key message might be the importance of keeping tight control over costs. The key message is how you want your audience to act or react in response to the theme. Your key message should be a concise sentence that's clear, positive, and memorable. And be sure to refer to it as you build your story to ensure you stay on track.

The next strategy is to plan the plot and the narrative arc. The plot is the series of events leading from A to Z. It's the path the character takes to get to the moral of the story. The narrative arc is the flow of how the situation unfolds along the way. The simplest way to build a story is using a three-act structure – act one is the beginning, act two is the middle, and act three is the end. Every act bolsters some aspect of the key message.

The third strategy is to plan the central conflict. The central conflict – sometimes called the core conflict – is the substance or meat of the story. It usually happens in act two of your story, which is where the main conflict comes into play. Here, you describe the conflict, the impact it has on the main character, and what's needed to overcome it.

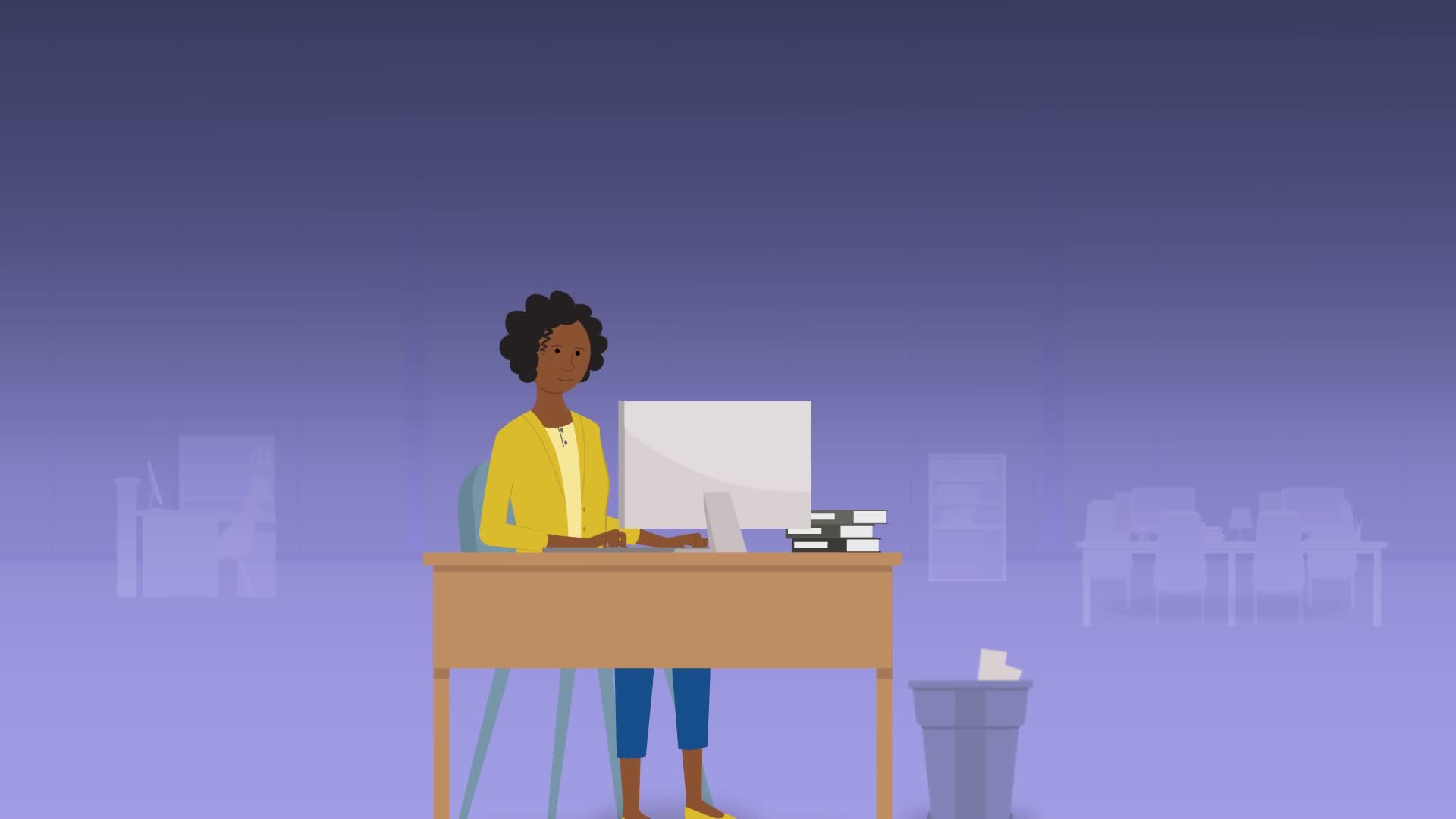
The tension that results from conflict maintains your audience's interest and keeps them engaged, so it's important to plan this section carefully. And then in the next act, the character steps up and overcomes the problem. That's where the key message is delivered or reinforced, showing the audience how to apply the lesson learned: maybe it's finding the courage to slay the dragon; maybe it's researching sound investments or buying the right toothpaste.

And the final strategy for structuring your story is to create an outline. The outline is where you sketch out the key details and lay out their context and flow. [Video description begins] *The following information is displayed on screen: GET EVERYTHING CAPTURED:.* [Video description ends] Outline the opening, the conflict, the barriers, and the resolution. Ensure that there's a clear main character and that other characters are named. List each act, the important events, and the specific story details.

When you've done all this, you're ready to write. Don't worry too much at this stage about getting everything captured perfectly. Just get it all on paper. Then you can read it and consider the themes and if the story is structured effectively. Is anything missing? What needs more detail? What needs less? Is it coherent and does it make sense? The outline is your roadmap, leading your story, and your audience, on its journey from beginning to end.

When we hear a great story, we don't always – or even often – give any thought to the planning and structuring that went into it. But every great story has been carefully plotted out. And your business story deserves no less effort.

## 7. Video: Telling Your Story (bs\_acm21\_a01\_enus\_07)

****

After completing this video, you will be able to identify strategies for effectively drafting and presenting a business story

* *identify strategies for effectively drafting and presenting a business story*

[Video description begins] *Topic title: Telling Your Story.* [Video description ends]

You have a killer business story to tell. You know the audience. You've plotted it out in detail. And you've worked out the structure. You're almost there... but not quite. There's still that one big step remaining. You still need to write and present it. And that's when strategies for effectively drafting and presenting business stories come into play.

It seems obvious, but the first step is to write and edit a draft of the story. The purpose of writing is clear, but editing and rewriting allow you to polish the story, smooth out the rough edges, and refine the action. It's about stripping out everything that doesn't need to be there.

When you're writing, remember, though, that telling a story isn't the same as just recounting the facts of a situation or event. You need to create interest and engagement. That means creating a world for your audience to explore. Let them know what it looks and sounds and feels like. And let them know the characters – how they think and feel about their world, where the conflict and tension comes from.

Most importantly, everything in your story should serve a purpose. If it doesn't, either cut it or change it. Keep the focus on your themes and key messages. And when you've written and edited... do it all again. Next, practice your presentation, making note of any issues. As the old saying goes, practice makes perfect.

When you've finished writing, read the story aloud – first to yourself and then to someone else. How does that come across? Is your message clear? Make a note of any issues, anything that doesn't sound quite right, or anything that doesn't quite work.

You need to find the right balance between being spontaneous and over-preparing. If you over-practice, you risk coming across as flat, like you're just reciting from memory. On the other hand, if you're underprepared, you risk sounding unfocused and unclear.

The third strategy is to make your presentation engaging. The way you deliver or present your story matters a lot – and we're talking about both the style and the method. Always keep the audience and goal in mind. There are certain tones that will turn an audience off. Don't be arrogant, don't vent about bad customers, and don't trash-talk your competitors.

Stay upbeat. If you're addressing a negative, keep the focus back on your positive solution or alternative. And make the story memorable. If it's appropriate, use humor and surprise to your advantage. If you're presenting verbally, use voices and body language to bring your characters to life. Then get your story out there. If you're telling it in-person to a live audience, record it so that you can share it later with others. An engaging story will get shared and retold, reaching an expanding audience.

And finally, make note of any lessons learned and adjust. Delivering or presenting your story isn't the end. It's time to take stock, learn lessons, and, if necessary, make some changes. It's easy to know if you've bombed: no one laughs; applause is polite, at best; attention wanders. Success is the opposite. If the audience responds the way you hoped, you've likely done well.

But even then, there's always room for improvement. It's important to review what worked and what didn't. For immediate feedback, ask your audience. If you've presented something online, watch for online reviews and comments. Based on the feedback you get, you may need to make further adjustments or remove some parts completely. Learn your lessons and apply them.

## 8. Video: Framing Your Story (bs\_acm21\_a01\_enus\_08)

****

After completing this video, you will be able to recognize basic story structures availble for business storytelling

* *recognize basic story structures availble for business storytelling*

[Video description begins] *Topic title: Framing Your Story.* [Video description ends]

We're exposed to stories every day. These stories are told in a variety of ways and through a multitude of methods. For instance, stories can be told verbally one-to-one or transmitted digitally through messages. Other stories are shared on social media, and we're inundated with ads – which are stories too – on TV and at the movies.

And what about the stories we share at work? We share stories during training sessions and staff meetings. And consider, too, the narratives that emerge over the course of e-mail threads – they're stories too. Most stories follow a certain structure or framework. There a lot of different story structures available, and these can be adapted and adjusted as the story requires, but their variety is not infinite. Most are variations on a theme, but that variation is good to be mindful of.

It helps to recognize the basic story structures available for business storytelling. Those structures can then be adapted to meet your story's need. For example, business origin stories are about the birth of an organization – how it was formed. A particularly interesting origin story helps your organization present itself as unique.

But further, the structure can be adapted to address the origin of a product, functional area, or specific team. Everything has an origin, and if what you have to communicate fits with the idea of the birth of something new, this structure likely fits. Then there are mission and value stories. These stories are about what your company and its people stand for.

A story about how your company puts people first, for instance, can really resonate with customers. Adapting it for your needs can help strike an emotional chord in your audience. And there are vision stories, which are about the future of the company and how its vision is becoming reality. Products, projects, and initiatives can all be said to fit into a company's fulfilling its vision.

Success stories are about beating the odds and meeting goals. Think about how different people define success and use that as inspiration. Don't focus solely on testimonials or financial success, but success in achieving business goals, meeting customer needs, and creating ways of being more effective, efficient, and productive are all valid stories.

There are also hurdle stories, which are a good way to build a reputation. These are stories about overcoming challenges. They inspire trust and confidence in your company, your project, or your team. "Look at how we overcame this obstacle," for example, is a way to structure an after-the-fact story, debrief, or lessons-learned.

And last, there are customer service stories. These can be personal or organizational. Make the customer your focal point. It's not about your product, it's about the customers who use it. A variation on the success story, customer service stories work nicely for external audiences and or any audience that you'd like to persuade to purchase or use your products and services.

Everybody has a story to tell. The success of yours is dependent on finding the right structure and method for telling it. They are a key tool for effective business communication.

## 9. Knowledge Check: Telling Business Stories

* *discover the subject areas that will be covered in this course*
* *recognize the benefits of transmitting information via storytelling*
* *identify the elements of an effective business storytelling effort*
* *recognize actions that create a connection with your audience, increasing the impact of your story*
* *recognize key strategies for the initial stages of planning a story*
* *identify strategies for outlining the beginning, middle, and end of a story*
* *identify strategies for effectively drafting and presenting a business story*
* *recognize basic story structures availble for business storytelling*

### **Question 1: Multiple Choice**

**What are the benefits of transmitting information via storytelling?**

#### **Options:**

1.

Stories help foster an empathetic relationship with the audience

2.

Stories enhance the perceived value of the subject

3.

Stories engage the audience on several different levels

4.

Stories help the audience better understand complex statistics

5.

Stories speak to both the left and right sides of the human brain

6.

Stories add credibility to marketing claims

#### **Answer**

1.

Stories help foster an empathetic relationship with the audience

2.

Stories enhance the perceived value of the subject

3.

Stories engage the audience on several different levels

5.

Stories speak to both the left and right sides of the human brain

#### **Feedback:**

*Option 1:*

*This option is correct. Stories help develop more intense connections between people, which, in turn, fosters empathetic relationships between storytellers and their audiences.*

*Option 2:*

*This option is correct. Beyond simple communication, telling the story behind a product can make consumers value and want that product more. As central as price or quality, an engaging narrative is often one of the main factors when people buy a product or service.*

*Option 3:*

*This option is correct. A good story engages the audience on multiple levels, including physically, mentally, emotionally, and spiritually. This makes the story more memorable for the audience.*

*Option 4:*

*This option is incorrect. This would not necessarily be an advantage of using storytelling to transmit information.*

*Option 5:*

*This option is correct. Storytelling engages every part of the brain. This means the listener can process the information and analyze it at the same time.*

*Option 6:*

*This option is incorrect. Although there are many benefits to transmitting information via storytelling, making claims seem more credible wouldn't necessarily be one of the benefits.*

### **Question 2: Multiple Choice**

**What are the elements of effective business storytelling efforts?**

#### **Options:**

1.

Controversy

2.

Tradition

3.

Tension

4.

Ease of understanding

5.

Novelty

6.

Relatability

#### **Answer**

3.

Tension

4.

Ease of understanding

5.

Novelty

6.

Relatability

#### **Feedback:**

*Option 1:*

*This option is incorrect. An effective business storytelling effort shouldn't be controversial or overly edgy. If there is a risk of causing offense or anything negative, it should be reconsidered.*

*Option 2:*

*This option is incorrect. Although a sense of tradition may occasionally be appropriate, a tired, clichéd, or overly-familiar narrative is not a good idea. Instead, there should be an element of novelty to ensure the audience is engaged and interested.*

*Option 3:*

*This option is correct. Having tension in a story will draw the audience in, making them engaged and focused. The story will hold their interest because they will want to know what's going to happen.*

*Option 4:*

*This option is correct. An effective business storytelling effort must be easy to understand. If the audience is struggling to understand the story, they won't be engaged in the story. Ease of understanding is key to ensuring the audience is engrossed and interested.*

*Option 5:*

*This option is correct. Although familiarity is good to a certain point, novelty is more engaging for audiences. A new story is more interesting and will draw the audience in.*

*Option 6:*

*This option is correct. One element of effective storytelling is relatability. People are more comfortable with things they are familiar with and can relate to. Relatability is a key factor in engaging audiences.*

### **Question 3: Multiple Choice**

**Which actions would help create a connection with your audience and increase the impact of your story?**

#### **Options:**

1.

Creating a generic story that appeals to as many people as possible

2.

Confidently presenting a complete list of product features with statistics, if available

3.

Getting to know and understand the target audience

4.

Crafting a story that resonates with the target audience

5.

Using the audience's own terminology and vocabulary when telling the story

6.

Drawing the audience in with an engaging story

#### **Answer**

3.

Getting to know and understand the target audience

4.

Crafting a story that resonates with the target audience

5.

Using the audience's own terminology and vocabulary when telling the story

6.

Drawing the audience in with an engaging story

#### **Feedback:**

*Option 1:*

*This option is incorrect. To create a connection with your audience and enhance the impact of your story, it's important to match your message to the target audience. It would be a mistake to try to create a generic story to appeal to everyone.*

*Option 2:*

*This option is incorrect. When the objective is to create a real, genuine connection, stories are essential. Statistics, straight facts, and product features alone are incapable of fostering connections.*

*Option 3:*

*This option is correct. The first thing to do when seeking to create a connection with your audience and enhance the impact of your story is to profile the target audience. That means finding out their likes, dislikes, habits, opinions, and priorities.*

*Option 4:*

*This option is correct. When trying to create a connection with your audience and enhance the impact of your story, you should match your message to the target audience. Instead of creating a generic message that has universal appeal, tailor the story to appeal to your specific audience.*

*Option 5:*

*This option is correct. When seeking to create a connection with your audience and increase the impact of your story, use the audience's own words. This makes it easier for them to understand and relate to the story, making a genuine connection more likely.*

*Option 6:*

*This option is correct. Stories can be used to connect more deeply with the audience. Unlike facts or product features alone, stories can create real, genuine connections between storytellers and their audiences. The more stories shared, the stronger the connection becomes.*

### **Question 4: Multiple Choice**

**What are key strategies for the initial stages of planning a story?**

#### **Options:**

1.

Ensure the moral of the story is stark and obvious

2.

Develop a main character to deliver the message

3.

Decide what value judgment you want the audience to make

4.

Establish the story's goals and objectives

5.

Forget about the audience for the time being

6.

Generate as many ideas as possible

#### **Answer**

2.

Develop a main character to deliver the message

3.

Decide what value judgment you want the audience to make

4.

Establish the story's goals and objectives

6.

Generate as many ideas as possible

#### **Feedback:**

*Option 1:*

*This option is incorrect. Although it's important that the moral of the story be clear enough that everyone can discern what it is, it's also important not to talk down to the audience or be patronizing.*

*Option 2:*

*This option is correct. During the initial stages of planning a story, you should create characters, including the main character or hero. This main character delivers the message of the story, and should represent the qualities of the brand or product.*

*Option 3:*

*This option is correct. You need to identify the moral of the story during the initial stages of planning. That means deciding what you want the audience to take from the story – the value judgment you want them to make about the brand or product.*

*Option 4:*

*This option is correct. One strategy for the initial stages of planning a story is to set goals for the story. That means determining the fundamental point of the story. What does it need to achieve?*

*Option 5:*

*This option is incorrect. It's important during initial planning to stay focused, which means keeping the story's goal and audience in mind. The objective and target audience are central to everything.*

*Option 6:*

*This option is correct. A key strategy during the initial stages of planning a story is to brainstorm ideas. It's always better to have as many ideas as possible. Brainstorming can bring out aspects of a story that you might have missed or overlooked.*

### **Question 5: Multiple Choice**

**What are effective strategies for structuring your story?**

#### **Options:**

1.

Sketch out the key details of the story

2.

Edit and rewrite any dialog that needs work

3.

Plan the central or core conflict of the story

4.

Plot the sequence of events and flow of the action

5.

Determine the fundamental point of the story

6.

Brainstorm ideas around the story's message

#### **Answer**

1.

Sketch out the key details of the story

3.

Plan the central or core conflict of the story

4.

Plot the sequence of events and flow of the action

5.

Determine the fundamental point of the story

#### **Feedback:**

*Option 1:*

*This option is correct. A key strategy for structuring a story is to create an outline, which is essentially a sketch of the key details. It should include the opening, the conflict, the barriers, and the resolution. It should also detail main characters.*

*Option 2:*

*This option is incorrect. Although editing dialog to get it right is important, it isn't part of structuring the story. Rather, it would be appropriate for a later stage in the story-writing process.*

*Option 3:*

*This option is correct. The central conflict, or core conflict, is essentially the substance of the story. Describe the nature of the conflict, the impact it has on the main character, and how it can be dealt with or overcome.*

*Option 4:*

*This option is correct. An important part of structuring your story is to plan the plot and the narrative arc. The plot is the chain of events that occur, while the narrative arc is how the situation unfolds along the way. The easiest way to do this is with a three-act structure: beginning, middle, and end.*

*Option 5:*

*This option is correct. When looking to effectively structure your story, you need to determine the key message – the fundamental point of the story. It's essentially how you want your audience to react to the story.*

*Option 6:*

*This option is incorrect. Brainstorming ideas is a useful strategy during the initial stages of planning the story, but not when you're working on the structure of the story.*

### **Question 6: Multiple Choice**

**Which strategies would be appropriate when drafting and presenting a business story?**

#### **Options:**

1.

Plot the sequence of events and flow of the action

2.

Consider lessons learned and make necessary changes

3.

Write, edit, and rewrite the story

4.

Ensure the style of presentation is engaging

5.

Create a well-crafted hero who represents the key message

6.

Read the story aloud to yourself and others

#### **Answer**

2.

Consider lessons learned and make necessary changes

3.

Write, edit, and rewrite the story

4.

Ensure the style of presentation is engaging

6.

Read the story aloud to yourself and others

#### **Feedback:**

*Option 1:*

*This option is incorrect. This is an appropriate strategy when structuring the story, but not when you're drafting or presenting it.*

*Option 2:*

*This option is correct. It's important to make note of any lessons learned when initially delivering or presenting the story and make any necessary adjustments. Even if the story is well received by the audience, there's always room for improvement.*

*Option 3:*

*This option is correct. The first thing to do is write and edit a draft of the story. The objective of this is to polish the story, smooth out the rough edges, and refine the action. The ultimate objective is to create a story that's interesting and engaging.*

*Option 4:*

*This option is correct. During the drafting and presenting stage, it's important to ensure that the presentation is engaging. This includes both the style and the method of the delivery. In particular, make sure the tone you're using is appropriate for the audience.*

*Option 5:*

*This option is incorrect. This is an appropriate strategy during the initial stages of planning the story, but not when you're drafting or presenting it.*

*Option 6:*

*This option is correct. It's important to practice your presentation and make note of any issues. A good way to do this is to read the story aloud, first to yourself and then to someone else. This will help you identify any issues or problems that need to be addressed.*

### **Question 7: Multiple Choice**

**What are the basic story structures for business storytelling?**

#### **Options:**

1.

Success stories

2.

Intrigue stories

3.

Customer service stories

4.

Regret stories

5.

Origin stories

6.

Vision stories

7.

Hurdle stories

8.

Value stories

#### **Answer**

1.

Success stories

3.

Customer service stories

5.

Origin stories

6.

Vision stories

7.

Hurdle stories

8.

Value stories

#### **Feedback:**

*Option 1:*

*This option is correct. Success stories generally look at how the organization overcame obstacles and achieved its goals.*

*Option 2:*

*This option is incorrect. An intrigue story is not one of the basic story structures for business storytelling.*

*Option 3:*

*This option is correct. Customer service stories are generally about the customers who use the company's products or services, with the customer as the focal point.*

*Option 4:*

*This option is incorrect. Regret stories aren't one of the basic business storytelling structures.*

*Option 5:*

*This option is correct. Origin stories generally address the beginning or origins of the organization.*

*Option 6:*

*This option is correct. Vision stories are generally about the future of the organization and how its vision is becoming reality.*

*Option 7:*

*This option is correct. Hurdle stories are about overcoming challenges. They're a good way to build a reputation.*

*Option 8:*

*This option is correct. Value stories are about what your company and its people stand for.*

## Course HTML Resources

| • | Glossary: Telling a Business Story |
| --- | --- |
|  |  |

| **arc** | Referring to the structure and shape of a story, it describes the flow of the story. It can be thought of as an extended storyline, detailing how things unfold along the way. |
| --- | --- |
| **empathetic relationships** | Relationships based on mutual understanding, respect, and support. By fostering an empathetic relationship, a business forges a closer connection with its customers. |
| **hurdle story** | One of the basic story structures for business storytelling, a story about overcoming challenges. It helps to build a company's reputation by inspiring trust and confidence. |
| **origin story** | One of the basic story structures for business storytelling, a story about the creation or birth of an organization. It helps the organization present itself as unique. |
| **perceived value** | Customers' assessment or rating of the value or merits of a product or service and its ability to satisfy their needs or expectations. |
| **plot** | The series of events that occur within the story from beginning to end. It can be thought of as the path the main character or hero takes to get to the moral of the story. |
| **values story** | One of the basic story structures for business storytelling, a story about what the company and its employees stand for. This type of story can resonate with customers, strengthening the connection between a company and its target market. |
| **vision story** | One of the basic story structures for business storytelling, a story about the future of the company and how its vision is becoming reality. This type of story can be inspiring. |

© 2025 Skillsoft Ireland Limited - All rights reserved.

| Job Aid: Key Steps in Planning, Outlining, and Writing a Business Story | |
| --- | --- |
| Effective storytelling within a business context requires thoughtful preparation and effective delivery. With practice, using storytelling can improve the way you communicate with key players inside – and outside – your organization. There are several strategies for plotting, structuring, and telling your story. | |
|  |  |
| Strategy | Details |
| Set goals for the story | Determine the point and objective of the storytelling effort. What are you trying to achieve? You need to understand what gap your story will fill and what connection it will make. |
| Brainstorm ideas | Brainstorming can bring out aspects of a story that you have overlooked or missed. Having lots of ideas is preferable to not having enough. |
| Create characters | Every story needs to have at least a hero or main character. However, there can be as many characters as necessary, but don't go overboard. Keep the characters believable and relatable. |
| Identify the moral | Identify the moral of the story by deciding what you want the audience to take from the story. That's the value judgment about your brand or the product you want the audience to make. |
| Determine the key message | Decide what the fundamental point of the story is, which likely derives from the central theme of the story. The key message is how you want your audience to act or react in response to the theme. |
| Plan the plot and the narrative arc | The plot is the series of events from beginning to end; it's the path the character takes to get to the moral of the story. The narrative arc is the flow of how the situation unfolds along the way. |
| Plan the central conflict | The central conflict – sometimes called the core conflict – is the substance or meat of the story. It usually happens in Act 2 of your story, which is where the main conflict comes into play. |
| Create an outline | The outline is a sketch of the key details, giving an overview of the opening, the conflict, the barriers, and the resolution. |
| Write and edit a draft | The aim of writing, editing, and rewriting is to polish the story, smooth out the rough edges, and refine the action. It's about stripping out everything that doesn't need to be there. |
| Practice your presentation | Practice makes perfect. When you've finished writing, read the story aloud – first to yourself and then to someone else – to ensure it comes across as intended. |
| Make your presentation engaging | The way you deliver or present your story matters, so make sure your presentation is interesting and engaging. Keep the audience and goal in mind. |
| Make note of lessons learned and adjust | After you have presented the story, it is time to take stock, learn lessons, and, if necessary, make some changes. Even if it went well and was positively received, there's always room for improvement. |

## TEST

Select the narrative frameworks that are widely recognized for structuring persuasive business stories and case studies.

**Result:** Partially correct. You needed to select **A, B, C** and **E.**

What you selected

How you did

* A.
* Answer A.
* Vision stories
* You selected this answer.
* You got it right.
* B.
* Answer B.
* Value stories
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Hurdle stories
* You selected this answer.
* You got it right.
* D.
* Answer D.
* Adventure stories
* You didn't select this answer.
* You got it right.
* E.
* Answer E.
* Customer service stories
* You didn't select this answer.
* You got it wrong.
* F.
* Answer F.
* Failure stories

Which strategies would contribute to the effective structuring of a story?

**Result:** Partially correct. You needed to select **A, B, D** and **F.**

What you selected

How you did

* A.
* Answer A.
* Determine the key message of the story
* You selected this answer.
* You got it right.
* B.
* Answer B.
* Determine the substance of the story and how it impacts the hero
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Make a note of lessons learned during the process
* You selected this answer.
* You got it wrong.
* D.
* Answer D.
* Create an outline with the key details of the story
* You selected this answer.
* You got it right.
* E.
* Answer E.
* Create engaging and believable characters
* You selected this answer.
* You got it wrong.
* F.
* Answer F.
* Map out what happens and how it happens
* You selected this answer.
* You got it right.

You are preparing a presentation that relies heavily on storytelling to influence your audience. Which actions are most likely to deepen audience rapport and amplify the story’s persuasive impact?

**Result:** Partially correct. You needed to select **A, B, D** and **F.**

What you selected

How you did

* A.
* Answer A.
* Ensuring you truly know the target audience
* You selected this answer.
* You got it right.
* B.
* Answer B.
* Using the audience's own words
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Using jargon and technical terms to convey authority
* You didn't select this answer.
* You got it right.
* D.
* Answer D.
* Matching the message to the target audience
* You selected this answer.
* You got it right.
* E.
* Answer E.
* Developing a story with universal appeal and relevance
* You selected this answer.
* You got it wrong.
* F.
* Answer F.
* Drawing the audience in with an engaging story
* You selected this answer.
* You got it right.

What are the advantages of using stories to transmit information?

**Result:** Partially correct. You needed to select **A, B, C** and **D.**

What you selected

How you did

* A.
* Answer A.
* Stories appeal to both sides of the human brain
* You selected this answer.
* You got it right.
* B.
* Answer B.
* Stories make the perceived value of the product or service seem greater
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Stories create strong bonds of empathy between people
* You selected this answer.
* You got it right.
* D.
* Answer D.
* Stories engage people physically, mentally, emotionally, and spiritually
* You didn't select this answer.
* You got it wrong.
* E.
* Answer E.
* Stories convince people to overlook or accept flaws or defects
* You didn't select this answer.
* You got it right.
* F.
* Answer F.
* Stories foster a sense of obligation and responsibility in audiences
* You didn't select this answer.
* You got it right.

When crafting a persuasive business narrative for senior stakeholders, which of the following practices would generally strengthen both the draft and the delivery?

**Result:** Partially correct. You needed to select **B, C, D** and **F.**

What you selected

How you did

* A.
* Answer A.
* Create an outline of the sequence of events and describe the conflict
* You selected this answer.
* You got it wrong.
* B.
* Answer B.
* Practice the presentation, making note of any issues
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Fine-tune the story, making necessary changes and adjustments
* You selected this answer.
* You got it right.
* D.
* Answer D.
* Ensure the delivery and presentation are engaging
* You selected this answer.
* You got it right.
* E.
* Answer E.
* Ensure the key message is appropriate for the target audience
* You selected this answer.
* You got it wrong.
* F.
* Answer F.
* Make note of lessons learned and adjust accordingly
* You selected this answer.
* You got it right.

Save and exit

A company is designing a storytelling approach to support a major change initiative. Which characteristics are essential for these stories to be effective in a business setting?

**Result:** Correct. Great job!

What you selected

How you did

* A.
* Answer A.
* Ease of understanding
* You selected this answer.
* You got it right.
* B.
* Answer B.
* Relatability
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Novelty
* You selected this answer.
* You got it right.
* D.
* Answer D.
* Tension
* You selected this answer.
* You got it right.
* E.
* Answer E.
* Security
* You didn't select this answer.
* You got it right.
* F.
* Answer F.
* Mysticism
* You didn't select this answer.
* You got it right.

During the concept and planning stage of a story, which of the following methods are suitable for structuring your narrative before you draft in detail?

**Result:** Partially correct. You needed to select **B, D, E** and **F.**

What you selected

How you did

* A.
* Answer A.
* Eliminating distractions by focusing on a single idea
* You selected this answer.
* You got it wrong.
* B.
* Answer B.
* Determining why you're telling the story
* You selected this answer.
* You got it right.
* C.
* Answer C.
* Creating bizarre, fantastical characters
* You selected this answer.
* You got it wrong.
* D.
* Answer D.
* Identifying the moral of the story
* You selected this answer.
* You got it right.
* E.
* Answer E.
* Brainstorming ideas and new angles
* You selected this answer.
* You got it right.
* F.
* Answer F.
* Developing a hero who represents the message
* You didn't select this answer.
* You got it wrong.

Save and exit

**28 -** [**Agile Principles and Methodologies | Codecademy**](https://www.codecademy.com/learn/ext-courses/agile-principles-and-methodologies)

Agile projects use short work iterations and incremental development of products that focus on business priorities and customer value.

In this course, you'll learn about Agile concepts that are fundamental when managing projects, including the four Agile values and twelve Agile principles. This course also covers the five phases of the Agile project management model, and introduces you to the most common Agile methodologies and frameworks. Finally, this course introduces key activities for managing Agile projects, including creating a product vision and project charter, and best contract and documentation types.

This course is one of a series in the Skillsoft learning path that covers the objectives for the PMI Agile Certified Practitioner (PMI-ACP)® exam. PMI-ACP is a registered mark of the Project Management Institute, Inc.

### **Duration**

43m 48s

43minutes 48seconds

### **Prerequisites**

Before taking this course a learner should have a general knowledge of terms and duties relating to managing a project.

### **Expertise Level**

Intermediate

Everyone

### **Code**

bs\_apj13\_a01\_enus

### **Training Credits**

PMI Professional Development Units (PDUs) -

1 credit

**To qualify for PDU credits:**

* **Watch all videos**

### **Objectives**

* identify characteristics of the Agile method
* distinguish between primary and secondary Agile values
* identify the five phases of the Agile project management model
* identify some of the methodologies that can be used for Agile project management
* recognize the four Scrum inspect and adapt events
* identify the five ADAPT steps required to transition to Agile
* identify the recommended components of a business case
* recognize the elements of a project charter
* identify the contract types suitable for Agile projects
* identify helpful types of documentation for Agile projects

# Agile Principles and Methodologies

Agile projects use short work iterations and incremental development of products that focus on business priorities and customer value.

In this course, you'll learn about Agile concepts that are fundamental when managing projects, including the four Agile values and twelve Agile principles. This course also covers the five phases of the Agile project management model, and introduces you to the most common Agile methodologies and frameworks. Finally, this course introduces key activities for managing Agile projects, including creating a product vision and project charter, and best contract and documentation types.

This course is one of a series in the Skillsoft learning path that covers the objectives for the PMI Agile Certified Practitioner (PMI-ACP)® exam. PMI-ACP is a registered mark of the Project Management Institute, Inc.

## Table of Contents

[1. Video: Agile Principles and Methodologies (bs\_apj13\_a01\_enus\_11)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_0)

[2. Video: Understanding Agile (bs\_apj13\_a01\_enus\_01)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_1)

[3. Video: Agile Values and Principles (bs\_apj13\_a01\_enus\_02)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_2)

[4. Video: Agile Project Management Model (bs\_apj13\_a01\_enus\_03)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_3)

[5. Video: Agile Methodologies (bs\_apj13\_a01\_enus\_04)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_4)

[6. Video: Scrum Framework (bs\_apj13\_a01\_enus\_05)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_5)

[7. Video: Adopting an Agile Approach (bs\_apj13\_a01\_enus\_06)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_6)

[8. Video: Initiating an Agile Project (bs\_apj13\_a01\_enus\_07)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_7)

[9. Video: Creating Vision and Charting a Project (bs\_apj13\_a01\_enus\_08)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_8)

[10. Video: Agile Contracts (bs\_apj13\_a01\_enus\_09)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_9)

[11. Video: Agile Documentation (bs\_apj13\_a01\_enus\_10)](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_10)

[12. Knowledge Check: Key Agile Concepts](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_11)

[Course HTML Resources](https://cdn2.percipio.com/secure/c/1763799075.b5f553928b4caee8233e850bf84910bfd11e8293/eot/transcripts/4f39d3cb-acdc-4f4d-b165-a08cad317103/bs_apj13_a01_enus.html#section_13)

## 1. Video: Agile Principles and Methodologies (bs\_apj13\_a01\_enus\_11)

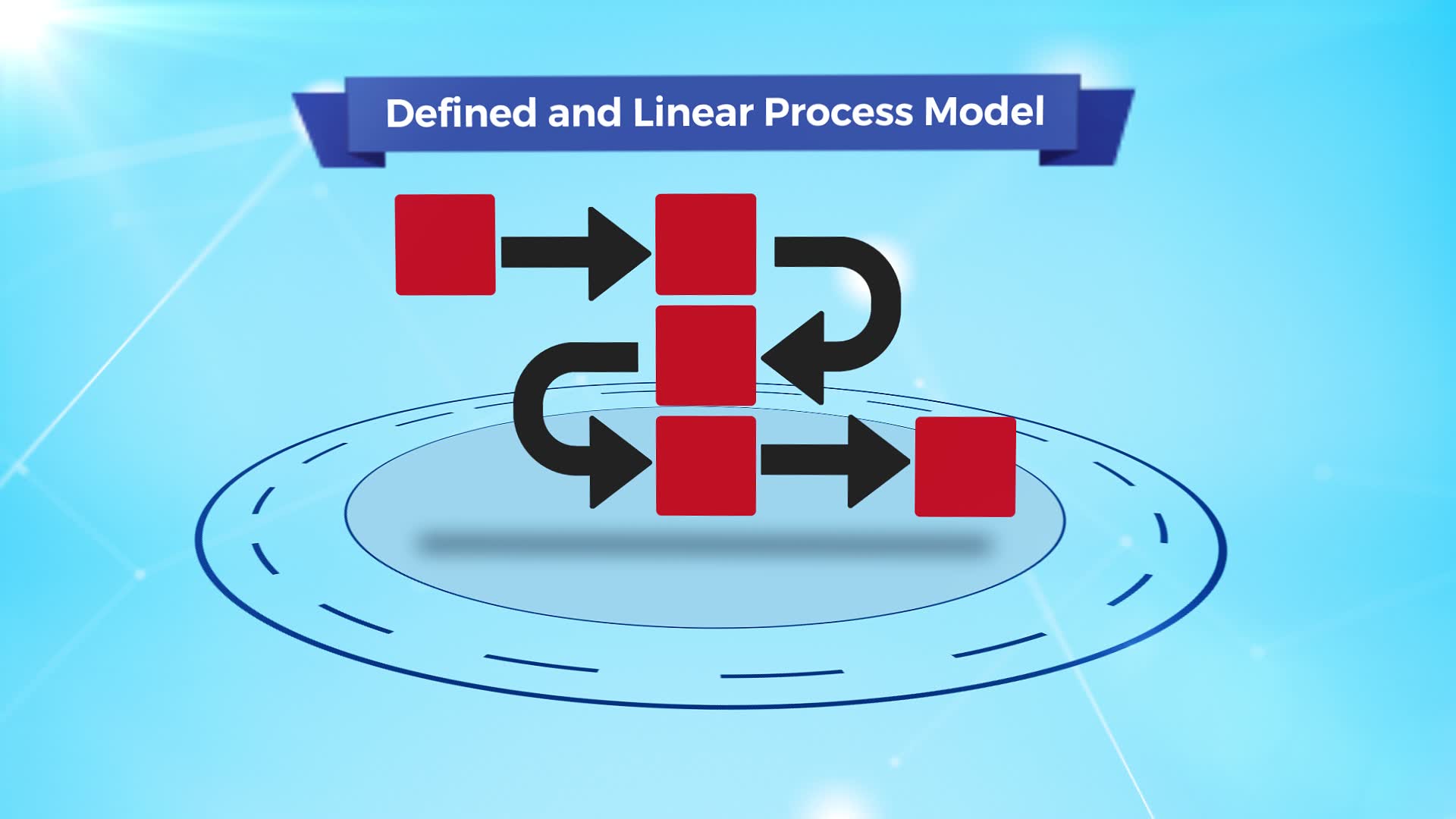
****

In this video, you will learn the key concepts covered in this course.

***No Objectives***

*[Course title: Agile Principles and Methodologies. The presenter is Barbara Waters, PMI-ACP.]* Agile Projects are characterized by the use of short work iterations and incremental product development. In this course, you'll learn about the core values and principles outlined by the Agile Manifesto. You'll also learn about some of the more common Agile methods and practices, how to create an Agile product roadmap, and the unique aspects of Agile contracts and documentation.

## 2. Video: Understanding Agile (bs\_apj13\_a01\_enus\_01)

****

In this video, you will learn how to identify characteristics of the Agile method.

* *identify characteristics of the Agile method*

*[Topic title: Understanding Agile.]* While there are countless ways you can manage a project, most common project management methodologies fall into one of two model types. Either a defined and linear process model, or more of an empirical and iterative process model.

In a defined and linear process model, there's a very linear approach planned for the project work. And work gets carried out following that plan through distinct and complete project phases. You complete one phase of a project, then move on to the next phase, each phase needs to be completed before moving to the next.

In an empirical and iterative model on the other hand, you use actual and empirical data gathered as project work is performed, working in iterations, building on what's been done, and introducing or changing elements as you go. Sometimes performing work in cycles to create the end result. *[A diagram of Empirical and Iterative Model displays. Based on empirical data gathered, or requirements, a project is built. Building a project includes the following three stages: Design and Development, Testing, and Implementation.]*

Agile Project Management is considered an incremental model for project management. So, instead of project work being completed in a linear model with one final delivery phase, your project work is divided up into increments. And each increment of project work goes through requirements, design, development, and then testing and delivery, before moving on to the next piece of project work. So you're taking it one piece at a time and delivering each piece.

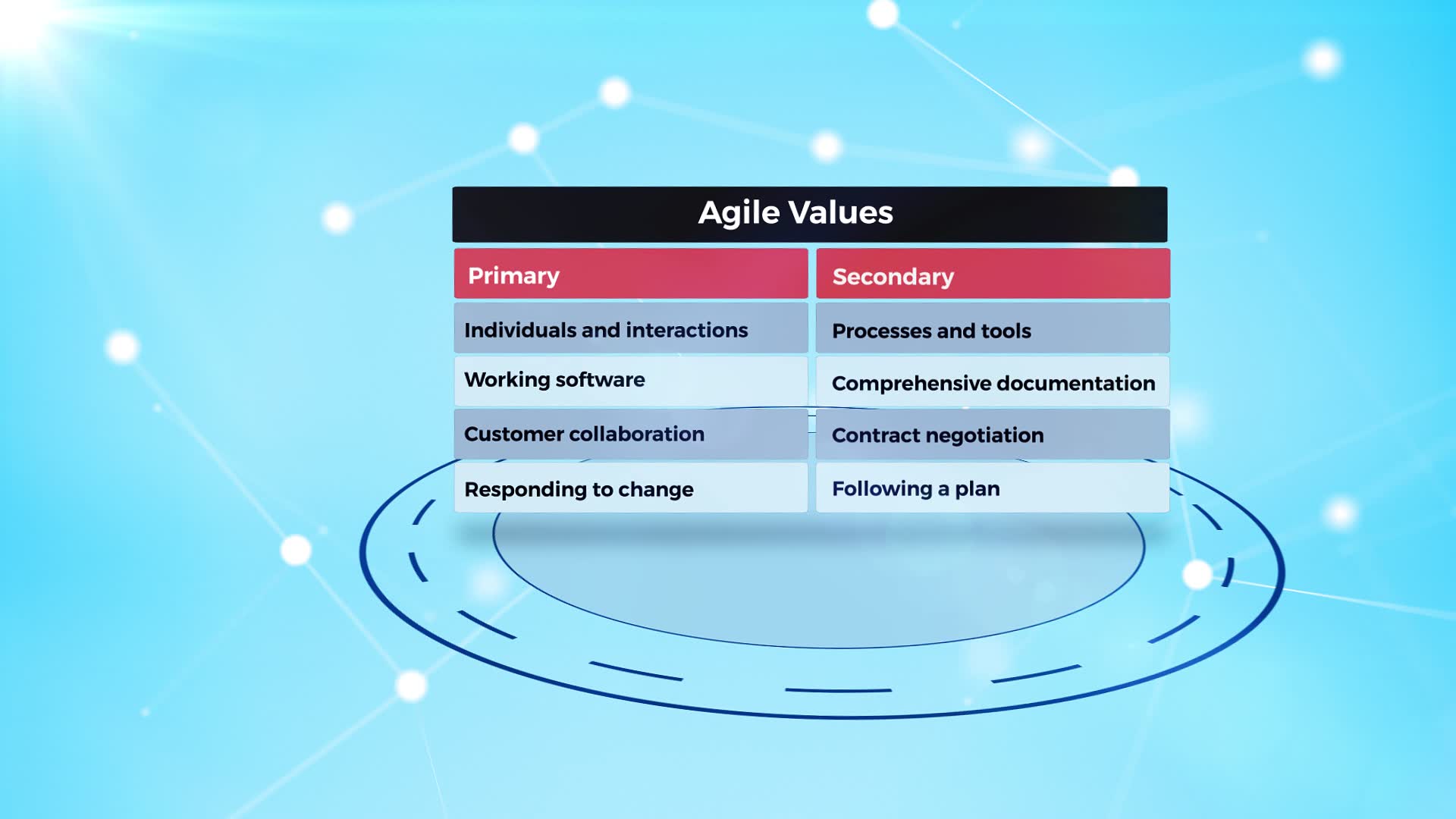
Agile Project Management is additionally considered an iterative model. So not only are you building the pieces of your final product incrementally over time, you're also iterating. *[An example depicting multiple iterations displays. Iteration 0 is Project Setup Plan, Iteration 1 is Plan, Develop, and Test Feedback, which is repeated as needed, and Iteration n is Develop and Test Release Product.]* Meaning in the first increment, you might initially build the minimum features that are required for your product, such as login functionality. *[An example showing login functionality displays. It has a username text box, a password text box, and a Sign in button.]* Then once you've tested that functionality, in the next increment of work you may build more robustness into that feature. *[A Forgot Password button is added to the login functionality.]* So you're incrementally adding different features to your end product, and adding robustness to your features with each iteration.

With Agile being iterative and incremental, the focus is on delivering the highest value items first. So if you're starting with highest value items first, then you know that you've at least tackled the things that are the most important to your stakeholders.

Another characteristic of Agile Project Management, is that you identify issues much earlier. Because testing is ongoing with each iteration rather than just in one testing phase at the end of the project life cycle, issues are realized much earlier in the development process. Feedback is also obtained early and often because at the end of each iteration, you're getting feedback from the project stakeholders. And then incorporating that feedback into the next iteration. Once that feedback has been received, and since you haven't moved too far ahead in the project work, it's a lot easier to make changes as well.

In summary, the Agile Project Management approach is incremental and iterative. Other characteristics of Agile are that it helps focus on the highest value requirements first. Issues get identified earlier in the project life cycle. Stakeholder feedback is received earlier and often. And changes are easier to implement.

## 3. Video: Agile Values and Principles (bs\_apj13\_a01\_enus\_02)

****

In 2001, a group of independent software practitioners came together and authored an Agile Manifesto in response to the need for a development methodology that was driven by the features and end product, rather than traditional project management methods. In this video, you’ll explore the primary values and secondary values of Agile, and how to distinguish between them.

* *distinguish between primary and secondary Agile values*

*[Topic title: Agile Values and Principles.]* So where does Agile come from, and what is it based on? Well, in February of 2001, a group of 17 independent software practitioners came together and authored what is called the Agile Manifesto.

The Agile Manifesto was born out of a need for a software development methodology that was driven by the features and end product, rather than being driven by traditional project management methods that focused more on the project management process itself, and less on the final product. The Agile Manifesto is comprised of 4 values and 12 principles.

The Agile Manifesto contains four values, which are further broken down into two sets, primary values and secondary values. *[The four Primary values are: Individuals and interactions, Working software, Customer collaboration, and Responding to change. The four Secondary values are: Processes and tools, Comprehensive documentation, Contract negotiation, and Following a plan.]* Essentially, all of the values contribute to the success of your project, but the primary values are the values that contribute most to successful projects, and are valued over the secondary values.

So the first Agile value says that we value individuals and interactions over processes and tools. Sometimes in our corporate environments or in team environments, we value certain processes and working with certain tools more than we value the individuals and interactions that create success. What that means is if we have a tool, for example, that we're using that costs us a lot of overhead, we should be thinking twice about using that tool. We should be thinking more about, how do we foster more communication? How do we foster more collaboration? And if there is a process that ultimately seems wasteful and doesn't really create a lot of optimization and people being creative or doing their best work, then we should rethink that process and focus more on having individuals interact in ways that bring out the best ideas.

The second Agile value says that we value working software over comprehensive documentation. What this means is working software really should be our measure of progress. The ability to start a project and quickly prototype something, or come up with something that works, is a lot more valuable than working for weeks and weeks, and in some cases, months, on documenting everything very comprehensively, especially for software development projects, since there's a lot of change to the final product across the life of the project.

The third value is that we value customer collaboration over contract negotiation. So if you've worked in an environment where you have a defined project plan, it sometimes becomes a big pain to actually have your stakeholders request a change. That change becomes a big contract negotiation. A change request goes in, and then you have to identify whether the change is worth it. In an Agile environment, it's understood that change is constant. Change is going to happen. And so the focus should be on collaboration with the customer to make sure that we understand the impact of the change, the need for the change, and the value that comes out of the change, instead of trying to negotiate whether or not to make changes.

The fourth and final Agile value is that we value responding to change over following a plan. This one is pretty self-explanatory. We need to be able to respond to change and have a system that allows us to do so dynamically, versus following a linear defined plan, where we try to define everything upfront, which is typically going to change over the course and lifetime of a project anyway, particularly in a software development project.

In addition to the four Agile values outlined in the Agile Manifesto, there are 12 principles. These principles should drive all work on your project. The first six principles are satisfy the customer, welcome change, deliver software frequently, work together, motivate individuals, and use face-to-face communication.

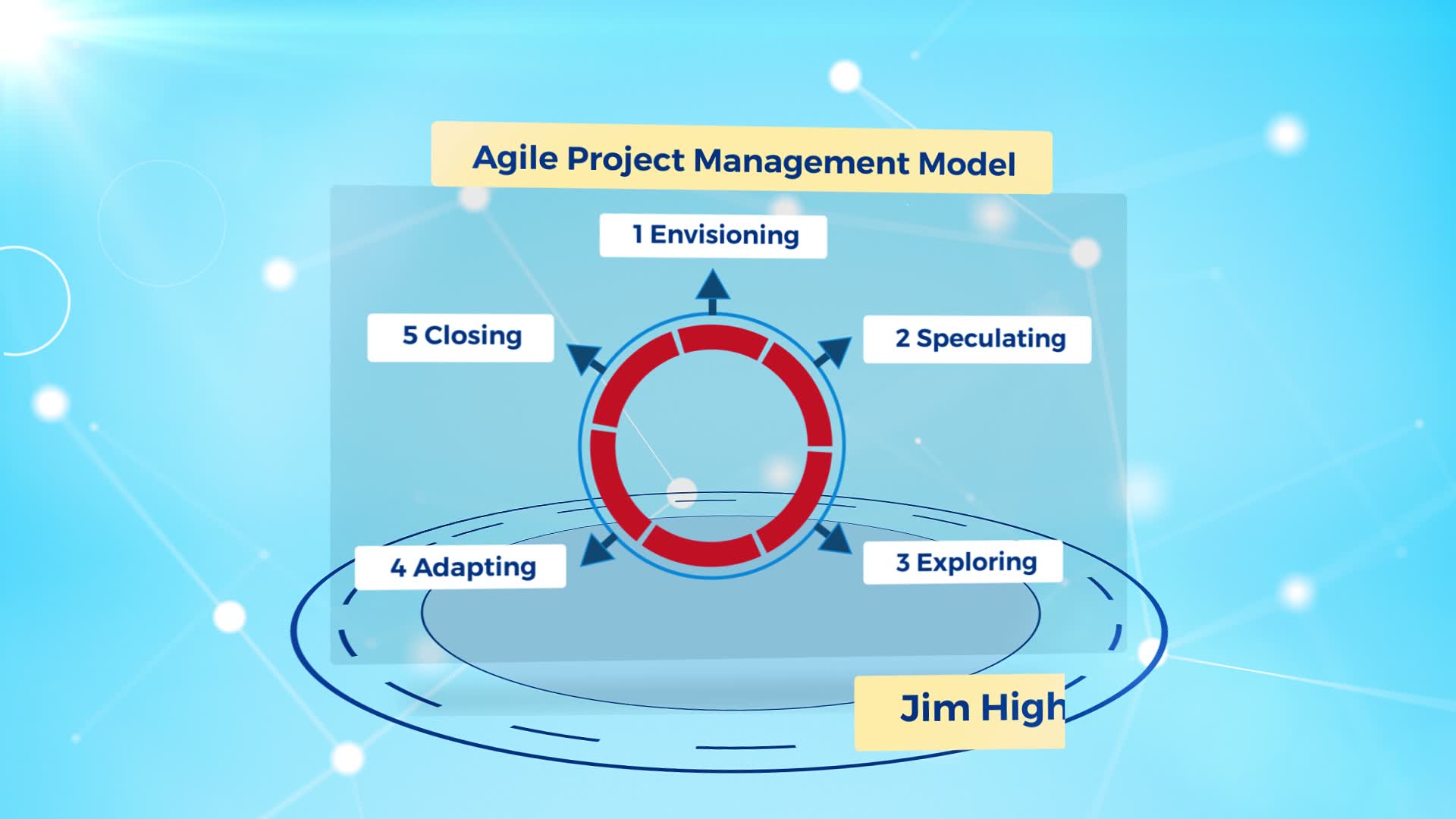
It is important to satisfy the customer. The highest priority is to satisfy the customer through early and continuous delivery of valuable software. It's also essential to welcome change, even late in development. Agile processes harness changes for the customer's competitive advantage. It's also important to deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter time-scale. And working together is vital. Businesspeople and developers must work together daily throughout the project. Motivating individuals is also key to your project's success. It's important to build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. And face-to-face communication is the most efficient and effective method of conveying information to and within a development team, particularly when time scales and development require the quick transfer of information.

The remaining six principles are that working software equals progress, constant pace, technical excellence, simplicity, self-organizing teams, and reflection. This means that working functionalities in the product are your signs of progress along the project timeline, not the succession of project phases in a spreadsheet. Constant pace is also needed. Agile processes promote sustainable development. The sponsors, development team, and users should be able to maintain a constant pace indefinitely. In addition, continuous attention to technical excellence and good design enhances agility. Simplicity is key. Keeping it simple is an art. Sometimes we tend to think that the more features we build or the more complexity we build into our features, the better, when actually, the best products are the simplest ones that have all the features and function the customer wants, but remain uncomplicated. In addition, the best architectures, requirements, and designs emerge from self-organizing teams. And finally, the principle of reflection brings success, and carries it through future work. At regular intervals, the team should reflect on how to become more effective, then tunes and adjusts working behaviors and procedures accordingly.

In summary, the Agile Manifesto outlines four values that should be kept in mind for project management. The values are, value individuals and interactions over processes and tools. Value working software over comprehensive documentation. Value customer collaboration over contract negotiation. And value responding to change over following a plan.

The Agile Manifesto also outlines 12 principles that should guide your project. They are, satisfy the customer, welcome change, deliver software frequently, work together, motivate individuals, use face-to-face communication, working software equals progress, constant pace, technical excellence, simplicity, self-organizing teams, and reflection.

## 4. Video: Agile Project Management Model (bs\_apj13\_a01\_enus\_03)

****

Learn how to identify the five phases of the agile project management model.

* *identify the five phases of the Agile project management model*

*[Topic title: Agile Project Management Model.]* While the Agile manifesto values individuals and interactions over processes and tools, and working software over getting caught up and focusing on comprehensive documentation, there is still a recognized value in a model outline for successful Agile project management.

Jim Highsmith, one of the authors of the Agile manifesto, developed the Agile project management model. The model consists of five phases, and it's worth noting that these five phases do not have a linear progression, but rather they are cyclical in nature. Agile is really a cycle and a set of iterations or loops where we're doing a lot of different activities within each cycle, rather than working through a prescribed path with distinct step.

The first phase of the Agile project management model is the envisioning phase. During this phase, you envision what this product is going to be, what type of value you should be delivering to your end customers, and what the vision is for the overall project or product.

The second phase is the speculating phase. In the speculating phase, you start thinking about how to implement different features or functionality that will actually fulfill that vision that you created in the envisioning phase.

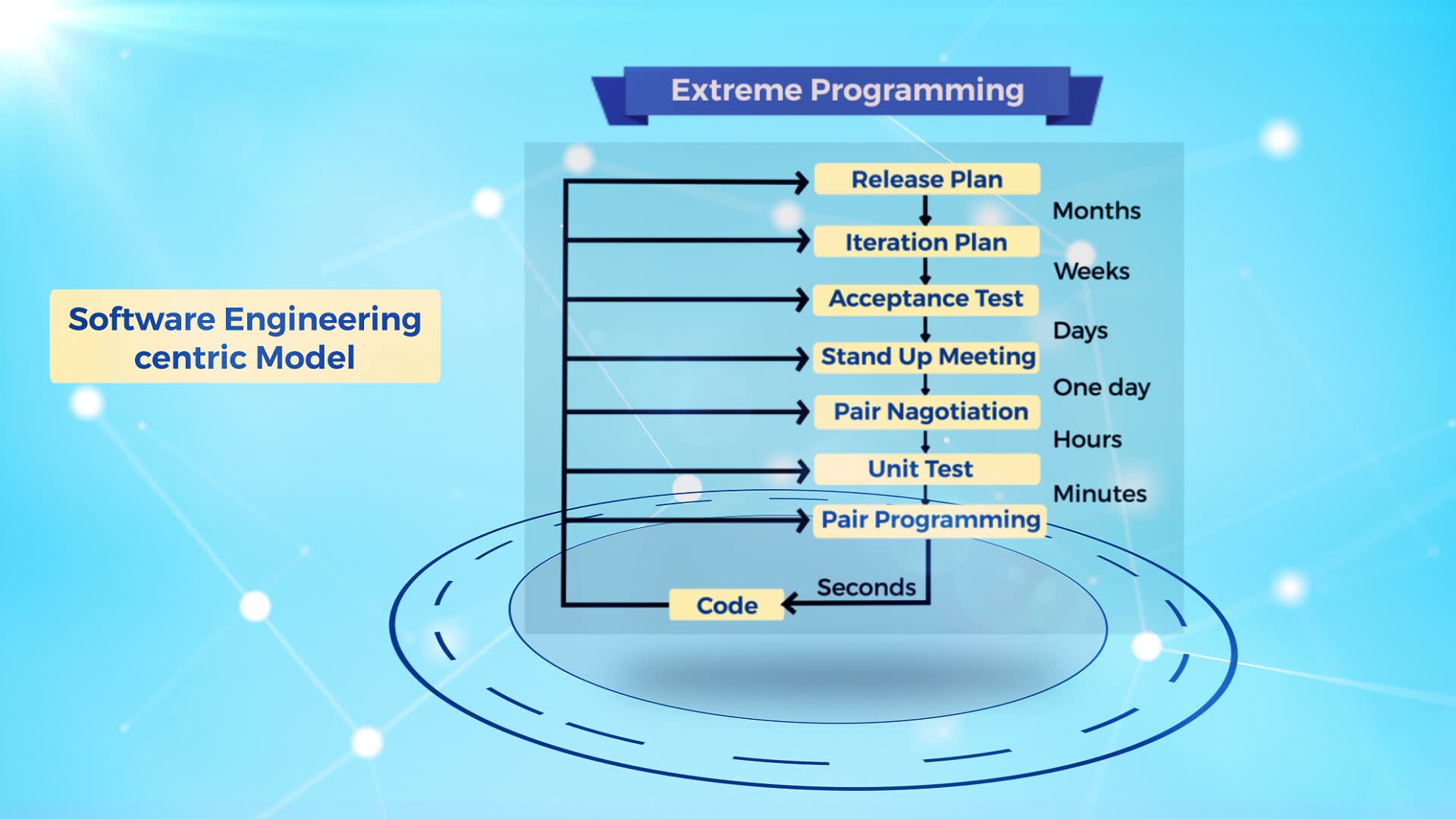
Next is the exploring phase. In the exploring phase, you're performing iterations of learning. This is the phase in which you're developing code, you're developing software, you're testing it, you're getting feedback on it, and you're figuring out the way to fulfill that vision.

The next phase is the adapting phase. In Agile, in every cycle, and in every iteration, there is adaptation. As you learn, you adapt and change the plan. You might change your idea of what is higher priority. You can even be changing the way you work in order to optimize and come out with the best ideas, as well as a really effective strategy for working together as a team.

And finally, the last phase is the closing phase. Because the Agile project management model is iterative, that could mean closing a specific iteration, or the whole project. *[An example depicting multiple iterations displays. Iteration 0 is Project Setup Plan, Iteration 1 is Plan, Develop, and Test Feedback, which is repeated as needed, and Iteration n is Develop and Test Release Product.]*

In summary, the five phases of the Agile project management model are, envisioning, speculating, exploring, adapting, and closing.

## 5. Video: Agile Methodologies (bs\_apj13\_a01\_enus\_04)

****

In this video, you will learn how to identify some of the methodologies that can be used for Agile project management.

* *identify some of the methodologies that can be used for Agile project management*

*[Topic title: Agile Methodologies.]* There are what seems like countless methodologies that you can use for managing your Agile project. You're specific project type will help determine whether an Agile methodology is suitable and which methodology should be used. You need to consider both criticality and safety and security requirements. You can implement an Agile model rigidly with no deviations, partially tailoring it to specific needs and in combination with another methodology as a hybrid model. Regardless of what you select, there are a number of common ones that you should be aware of and consider for your Agile project.

The first is the extreme programming, or XP methodology. XP is one of the first frameworks that could be considered Agile. *[The iterations involved in Extreme Programming methodology are: Release Plan, Iteration Plan, Acceptance Test, Stand-up Meeting, Pair Negotiation, Unit Test, and Pair Programming.]* XP is a software engineering centric model and was developed by a number of software engineers who wanted to improve the way they worked together as teams, the way they worked with their stakeholders and the quality of the end product that they were building. XP focuses on the ongoing rapid delivery of software through quick, short releases with a recommended one week iteration. Each iteration results in production ready code.

Lean principles and tools are another approach to Agile project management. Lean principles and tools don't prescribe specific development methods, but instead provide guidelines for streamlining the development process. *[The various Lean Principles are: Identify Value, Map the Value Stream, Create Flow, Establish Pull, and Seek Perfection.]* The guidelines are driven by seven key principles to accomplish this, eliminate waste, build quality in, create knowledge, defer commitment, deliver fast, respect people, and optimize the whole.

Kanban, another Agile framework, is derived from Lean principles and tools. Kanban principles include visualizing a workflow, limiting work in progress or WIP, focusing on the workflow and continuous improvement.

Crystal Methodologies are a family of Agile methodologies named after the colors of crystals of different hardness. *[A sample diagram of Crystal Methodologies displays.]* The Crystal Methodology you should apply depends on the complexity or hardness of a specific software development project. As the size and criticality of the project increases, a methodology with more prescribed steps and artifacts is required to keep control of the development process.

Feature-driven Development, or FDD, provides a client-centric, architecture-centric and pragmatic software development process. All aspects of the software development process are planned, managed, and tracked at the level of individual software features. Five main activities in FDD are develop an overall model, build a features list, plan by feature, design by feature and build by feature.

The Dynamic Systems Development Method, or DSDM is another framework for Agile development. It reflects a business perspective rather than a technical one and has three phases, pre-project activities, project activities, and post-project activities. DSDM requires a large number of artifacts and designates a large number of roles, but has flexibility in the way activities are incorporated. *[A diagram depicting Dynamic Systems Development Method displays. In the Pre-project phase, the initiation of the project takes place. Feasibility is a short phase to assess the viability and the outline business case. Foundation is a key phase for ensuring the project is understood and defined well enough so that the scope can be baselined at a high-level and the technology components and standards agreed, before the development activity begins. Exploration is an iterative development phase during which teams expand on the high-level requirements to demonstrate the functionality. Engineering is an iterative development phase where the solution is engineered to be deployable for release. In the Incremental Deployment phase, for each increment of the project the solution is made available. The Post-project phase assesses the accrued benefits.]*

Model-driven Development, or MDD, requires extensive models prior to production. The Agile version of MDD, or AMDD involves creating incremental models just to add sufficiently detailed support throughout the project life cycle. A team can begin by developing a high-level model of the project requirements, and then develop more low level models for each iteration requirement. AMDD principles include assuming simplicity, modeling with a purpose, valuing content above representation, creating quality work, communicating openly and traveling light.

Disciplined Agile delivery, or DAD, is a process decision framework that is a people first, learning-oriented hybrid Agile model. It has a risk value delivery cycle and it is goal-oriented, enterprise-aware and scalable. DAD is not prescriptive. Rather, DAD provides the guidelines for fitting together appropriate strategies and approaches for a successful outcome.

Test-driven Development, or TDD is a design technique that emphasizes unit testing with tests written before code. TDD components include test cases, test fixtures, test suites and test harnesses. TDD is useful for specification and validation rather than overall system design. *[A flowchart depicting Test-driven Development displays. The idea is that before adding a functionality, you write an automated test about how this new functionality that you want to introduce to your system (new code) has to behave. You then run the test and you wait for it to fail. You then again update the functional code to the specification, you write the simplest code, for it to pass. You run the test again and you make sure that this time it passes. At the end you have to refactor (even the simplest code has undesirable properties) and make sure that you have a clean code.]*

And finally, Behavior-driven Development, or BDD is often iterative and aims to improve a team's ability to deliver prioritized, verifiable business value using a shared language known as ubiquitous language.

In summary, there are numerous Agile methodologies you can use for your project. Some of the most common ones include, Extreme Programming or XP, Lean Principles and Tools, Kanban, Crystal Methodologies, Feature-driven Development, or FDD. Dynamic Systems Development Method, or DSDM, Agile Model-driven Development, or AMDD, Disciplined Agile Delivery, or DAD, Test-driven Development, or TDD, and Behavior-driven Development, or BDD.

## 6. Video: Scrum Framework (bs\_apj13\_a01\_enus\_05)

****

During this video, you will learn how to recognize the four Scrum inspect and adapt events.

* *recognize the four Scrum inspect and adapt events*

*[Topic title: Scrum Framework.]* As a lightweight framework for Agile project management, we say that Scrum is a framework versus a methodology since Scrum does not prescribe how to implement certain things such as reporting, source control, or even code. Scrum utilizes an iterative, incremental approach which allows for predictability and better risk management. So, Scrum is centered around iterations or sprints that are typically anywhere between one to four weeks in length. Most commonly these days, many teams are implementing a two-week sprint length.

Each sprint or iteration development period has a clear goal consisting of an agreed set of work items to implement during that iteration. So before starting any sprint, the team gets together and agrees on what work items they will complete within the course of a sprint. And at the end of each sprint, the goal is to have some kind of product increment that can be inspected and adapted, gaining feedback from stakeholders.

Each successive sprint then builds on the last. And planning occurs in between the sprints. So, although we do have a general high-level plan for the overall release, sprints also allow us to inspect and adapt and potentially change the plan for the next upcoming sprint based on feedback or based on something we've learned during the course of a sprint.

Inspecting and adapting are central to the success of Scrum. And Scrum includes four events that provide opportunities to inspect and adapt. These are sprint planning, daily Scrum, sprint review, and sprint retrospective.

The sprint planning meeting is the initial meeting where the team will commit to a set of deliverables for the sprint. The sprint planning meeting is an opportunity to inspect and adapt because at this meeting, the product owner is able to provide the team with some feedback based on either the last sprint or based on things that have changed since the last sprint. It's an opportunity to potentially reorder or reprioritize the requirements backlog.

The daily Scrum is another opportunity to inspect and adapt as a team. Everyday there's a Scrum meeting, also known as a daily stand up. Typically 15 minutes maximum in which team members discuss what they've done, what they are going to work on and any roadblocks or impediments to their work. Each team member answers three main questions. What work have I completed since the last Scrum and why? What do I plan on completing between now and the next Scrum? And do I have any roadblocks or problems that the team can help me overcome? On many occasions, the why part of the first question is skipped since everybody should know why they've worked on specific items. Those items typically line up with sprint goal and have been agreed on at the beginning of the sprint. However, in the case that something has delayed or changed priorities, it's good to make sure that everybody understands why priorities changed with work items and what challenges are being experienced?

The sprint review, which happens at the end of a sprint, allows the team to review and demo the product of their work to their stakeholders and to the product owner. It's also another opportunity to inspect their deliverables, to inspect what they've been working on, and adapt if necessary.

And lastly, the sprint retrospective is the fourth event that provides opportunities to inspect and adapt. The sprint retrospective is a meeting in which the team can get together and talk about what went well, what didn't go well, and what action items they can take in order to improve as a team.

In summary, the primary goal of the Scrum framework is to have opportunities to inspect and adapt as work progresses in your Agile project. The four Scrum events that provide opportunities for this are, sprint planning, the daily Scrum, the sprint review and the sprint retrospective.

## 7. Video: Adopting an Agile Approach (bs\_apj13\_a01\_enus\_06)

****

Discover how to identify the five steps required to transition to Agile.

* *identify the five ADAPT steps required to transition to Agile*

*[Topic title: Adopting an Agile Approach.]* When considering the adoption of an Agile approach, there are some factors to consider. The organizational structure, the project type, a customer's role, and the team composition all influence how Agile can be adopted. It's also important to consider how an organization approaches change. An organization that focuses on controlling change to minimize potential disruption may struggle with Agile methodology. If the focus is on welcoming change, an Agile methodology will be easier to support.

Regardless of the individual factors within your organization, the ADAPT steps developed by Mike Cohn identify five necessary requirements for successfully transitioning to Agile. These five steps are create awareness, increase desire to support change, develop ability, promote successes, and transfer the Agile mindset throughout the organization.

The first step of the ADAPT process is awareness. Your organization needs to be aware of the problems that it has currently. What are some of the process issues that are occurring? What are some of the inefficiencies? What are some of the issues that we want to improve upon? Also, awareness around how Agile methodologies can help us achieve those goals or fix those issues is key knowledge to have.

The second step is desire. There definitely needs to be a desire from the organization, or from enough people in the organization, to make the change and to make the shift towards Agile methodologies.

Ability is the next step. The actual ability for an organization to support an Agile approach is vital. In some cases, organizations may not have the ability to change because of certain factors. There might not be enough co-location of teams, or the engineering structure may not allow for things like test-driven development or unit testing. It could also come down to an organization just not currently having people with the right skill sets to adopt Agile.

Next, after deciding whether you have the ability to actually move to an Agile methodology, you want to start promoting it. This includes promoting Agile across the organization, not just specific to your project. And it also includes talking about the benefits that you expect to gain.

And the fifth and final step is transfer. This is the step where you finally make the transition and start implementing Agile methodologies or frameworks in your organization.

Moving to Agile is definitely not an easy thing to do. It includes change and requires successful change management. It includes a lot of understanding of why you're moving, and why you're trying to change the way that you've done things. It involves cooperation with governance, constraint and feedback, and organization and learning. However, the benefits for organizations that have moved to Agile have been very clearly measured, and show a lot of improvement in areas that people and organizations care about.

In summary, the ADAPT steps are required when transitioning to an Agile approach. The steps include awareness, desire, ability, promote, and transfer.

## 8. Video: Initiating an Agile Project (bs\_apj13\_a01\_enus\_07)

****

In this video, you will learn how to identify the recommended components of a business case.

* *identify the recommended components of a business case*

*[Topic title: Initiating an Agile Project.]* When initiating any project, it's a good idea to first make sure that you understand the opportunity and whether it makes sense for your business to pursue that opportunity. So the key activity for initiating an Agile project is establishing a business case or business justification.

The purpose of having a business case is multifold. A business case helps ensure that everyone understands what they are trying to achieve. By establishing a business case for the end product, it becomes very clear across the organization from executives to managers, to marketing, to sales, to the development team, what the goal is for developing this product or working on this project.

The business case also helps simplify decision-making by setting clear objectives and parameters. By understanding those parameters, you can also start to create a framework for making decisions around the product. A business case also helps keep the project team focused on meeting the customer's overall objectives for a project. By defining the value that the customer sees or that the customer is seeking, it becomes a lot easier for the team to focus on those things, versus perhaps features or functionality that the team considers as valuable.

Finally, the business case provides the key criteria for judging the success of a project.

So what exactly is included in a business case? Recommended components of a business case include an opportunity, goals, strategy, project vision, milestones, investment, and expected payback. Some components of a business case can include an opportunity. So an opportunity is a chance to create value or meet a business need. For example, in a hypothetical project, the stated opportunity might be to update the website with the company's revised corporate image, to improve functionality, and bring it in line with industry standards.

Another component in a business case are the goals which provide the reason why a project should be implemented. They also inspire a team and they need to be specific, measurable, attainable, and realistic.

Strategy is another component that should be included in a business case. Strategy is the plan that will help the customer achieve the specified goals.

The project vision, which specifies what the customer hopes a project will have achieved once it's completed, should also be included in the business case. The project vision also helps an Agile team understand the purpose of the project.

Milestones, which identify significant points in the development and shipping of a project or product, should be in your business case. Milestones help focus the team and make it easier to track progress.

The investment specifies what a customer will need to invest in a project based on each milestone.

And finally, the expected payback is included. The expected payback identifies both the financial benefit or return on investment that a customer can hope to gain from a project. As well as what the payback is for the company as a result of the project.

In summary, there are seven different components that should be identified in a business case when you are initiating an Agile project. They include opportunity, goals, strategy, project vision, milestones, investment, and the expected payback for both the customer and the organization.

## 9. Video: Creating Vision and Charting a Project (bs\_apj13\_a01\_enus\_08)

****

During this video, you will learn how to recognize the elements of a project charter.

* *recognize the elements of a project charter*

*[Topic title: Creating Vision and Charting a Project.]* Any project requires directions to steer the team toward success. Creating the product vision and developing the project charter are two key activities that help guide the project team to a accomplishing the goals of your Agile project.

The product vision is a compelling statement about why you're building a product, the benefit the product brings, who you're building it for and why you are uniquely positioned to develop it. A product vision also describes how a project can capitalize on the opportunities and fulfill the goals of the business case. A product vision should provide all the stakeholders, including the development team, with a common understanding of what's required, without limiting that team's creativity in finding solutions.

An important part of creating a product's vision is to interview the people who are stakeholders of the product about their vision for the product. The interview should focus on establishing minimum and maximum requirements for the product and investigate the relationship between the project's potential risks and benefits. The vision statement doesn't just describe a product's features, it focuses on how a product will add value. It should motivate the development team and help them envision the final product.

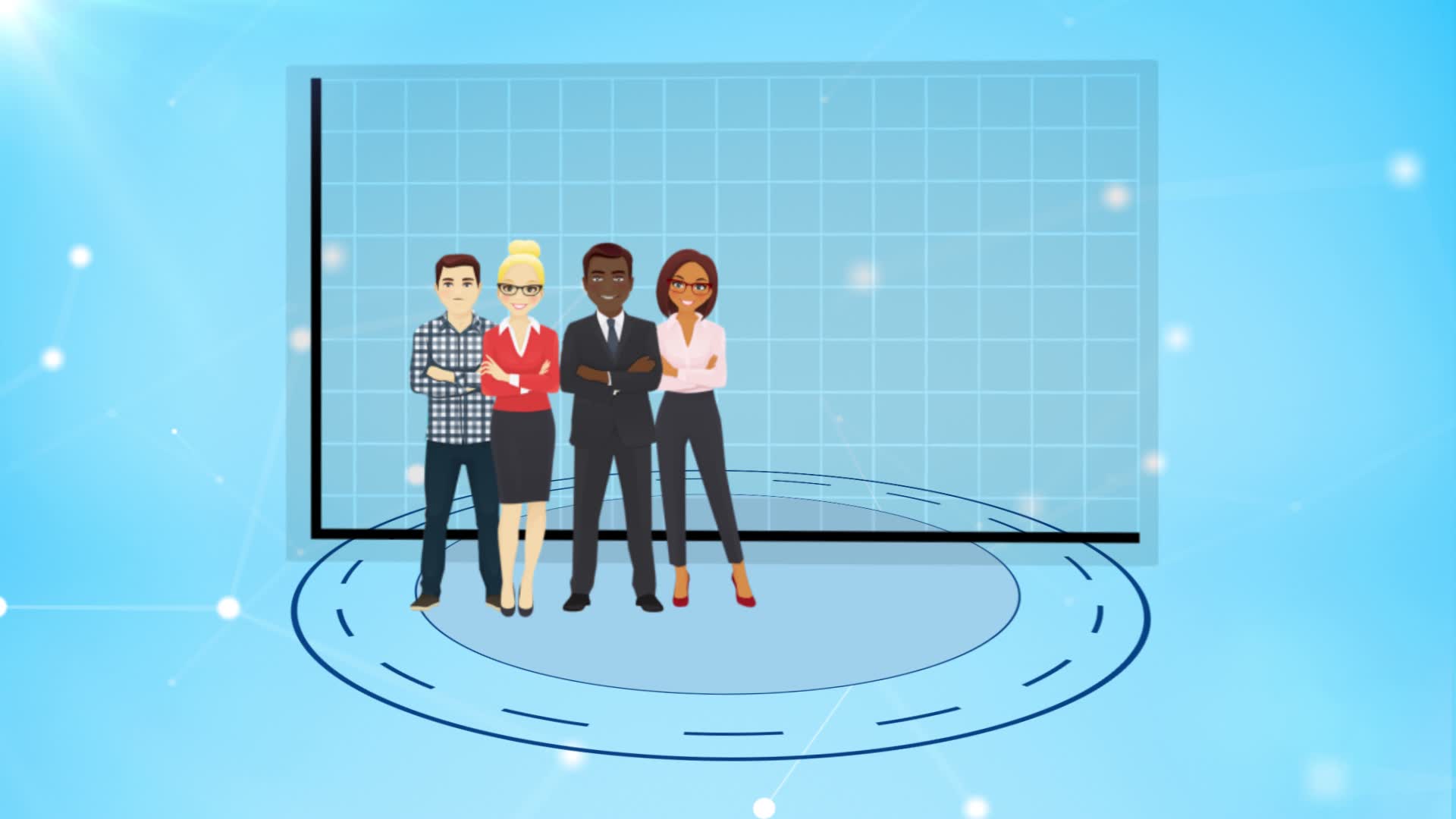
Project charters are great way to ensure that the intentions, outcomes and priorities are clear to both the stakeholders and the team. It's a good way to try to establish alignment on key elements of a project, especially early on. A project charter can contain many elements. But one of the most common ones is the vision statement, which defines the why of the project. This is the purpose, or the reason, why this project exists. The objectives or mission are the what of the project, and these state what will be done in the project to achieve the vision. It should also identify who the customer is and what problem is going to be fixed with the end product.

The success conditions, or success criteria, are a set of outcomes or results that define success of the project for management or stakeholders. A charter can also contain priorities and compromises. It's a good idea to understand what a stakeholder will and will not accept as a compromise before starting out on a project. And it's also important to know what risks the project has. As well as to think about the acceptable workarounds. Risks and mitigation strategies should be included in the project charter. Team roles should also be included. Who will be doing what should be documented, along with a high-level overview of what each person is responsible and accountable for.

The most important thing to note about a project charter in an Agile environment is that it's a one page document. It needs to include these high-level essential items that talk about what creates success for a project. But it shouldn't go on and on for many, many pages.

In summary, a project charter should be no more than a page long. But should still include the following elements. A vision statement, objectives or a mission of the project, customer, problem, success conditions or criteria, priorities and compromises, risks and mitigation and team roles.

## 10. Video: Agile Contracts (bs\_apj13\_a01\_enus\_09)

****

In this video, find out how to identify the contract types suitable for Agile projects.

* *identify the contract types suitable for Agile projects*

*[Topic title: Agile Contracts.]* Contracts help organizations manage their risks and resources. In traditionally managed projects, the most commonly used contract is a fixed-price contract.

Fixed-price contracts aren't generally the best for Agile projects, as they assume a fixed scope. They also place risk on the team implementing the project. Because if the work ends up exceeding the estimate or if requirements change, which they frequently do, the team has to bear that loss and still continue with the promised scope. Fixed-price contracts may also take focus away from what's best for the product and customer. And instead, developers become bound by legal limitations that are no longer in the customer's best interest.

In theory, a possible solution is for a contract to specify the overall business goals that a project team must enable a customer to meet. Instead of specifying detailed technical requirements. However, this isn't always practical. Because in terms of legal requirements, business goals need to be defined in monetary terms, which can be difficult.

So what types of contracts are suitable for Agile projects? You can use different contract types for each of the Agile development phases. So for example, in the initiation phase, you may use a service contract. Instead of measuring a specific deliverable like product development, a service contract may measure the time, cost, and materials spent on coming up with for example, a roadmap or a story map within that project.

The next phase, or the development phase, may be a series of fixed-price contracts for each iteration. Some variants on the fixed-price contract include fixed-range contracts and fixed-price per user story.

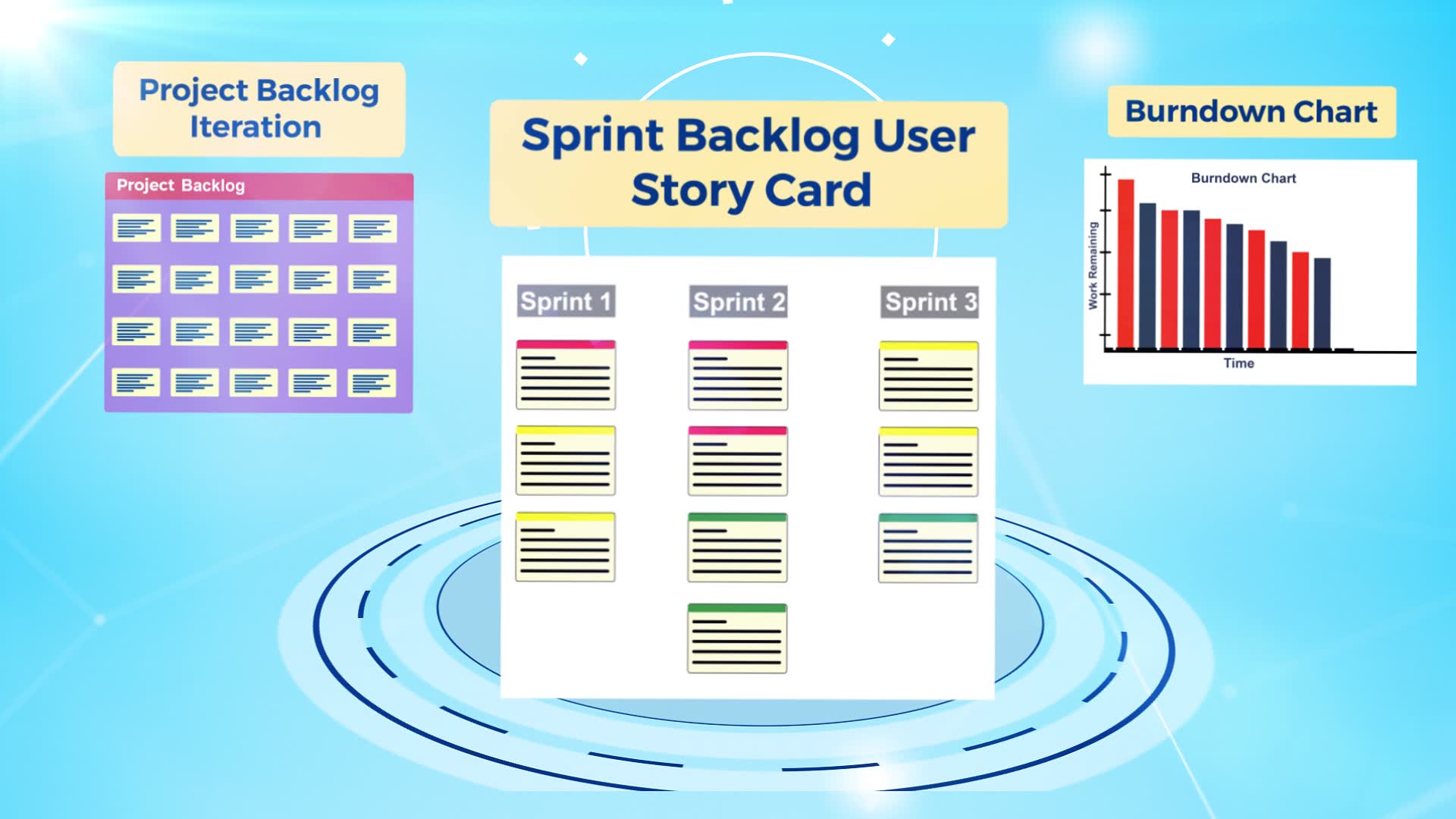
Cost-reimbursable or time-and-materials contracts operate on a pay-as-you-use basis. So it's in the client's hands to decide whether or not to pay the team to spend more time or more resources on developing additional requirements, making changes or enhancements, or fixing issues.

A not-to-exceed with fixed-fee contract can protect both the development team and the customer. It's made up of a fixed-fee base as well as a not-to-exceed condition. So it may be, for example, that our fixed-fee base for this project is x dollars and then the not-to-exceed condition is we will not exceed 15% of that base.

An incentive contract can also be used and can motivate a development team by offering rewards for good performance. The types of incentive contracts include, fixed price with incentive, cost-reimbursable with award fee, percentage increase or decrease based on time, and a fixed price user story with additional hourly rate.

In summary, while regular fixed-price contracts are not suitable for Agile projects, many other types of contracts are. They include service contracts for the series of fixed-price contracts, cost-reimbursable or time-and-materials contracts, not-to-exceed with fixed-fee contracts, and incentive contracts.

## 11. Video: Agile Documentation (bs\_apj13\_a01\_enus\_10)

****

Learn how to identify helpful types of documentation for Agile projects.

* *identify helpful types of documentation for Agile projects*

*[Topic title: Agile Documentation.]* One of the key principles of the Agile Manifesto is to emphasize Working Software over Comprehensive Documentation. So that means we prefer having Working Software be our primary measure of progress versus Comprehensive Documents. Agile approaches focus on keeping documentation light or having just enough documentation to help you move forward and make progress. Most agile teams use specific forms of documentation. And they favor those that are simple and highly visual.

So for example, a Project Backlog Iteration, Sprint Backlog User Story Cards, and Burnup and Burndown Charts are very common documentation forms used in Agile teams. All of these are very simple to use and are highly visual, and it's shown that visual communication is a lot more effective than communication via pages of text.

Sometimes people feel that documentation is required for every process in every project. Reasons might be that it's needed for Managing risk, Following a process, Information sharing or Detailing requirements. These are not valid reasons for generating comprehensive documentation, and can actually hinder, rather than help in Agile project success.

In an Agile environment, you manage risk better by focusing on delivering the highest value items first, versus by creating very comprehensive documents. And you can still follow a very specific and very effective process without having comprehensive documentation. In addition, sharing information among teams can happen in collaborative environments where people are talking verbally and communicating without the need for documents. And from an Agile perspective, thoroughly detailing requirements early in a project is wasteful. It's too soon at that point for customers to know what they really want at a detailed level. So requirements are likely to change.

We know that early on in a project is the point where we know the least that we are ever going to know about that project. That's the worst point in the project life cycle to try and create very detailed comprehensive documents about what's required.

With all of that being said, there are of course valid types of documentation that are quite useful and contribute to the success of an Agile project. For example, the Vision Statement defines the ultimate goals of a project and the why. Why we're building this project.

A Project Overview summarizes critical information about a project such as the Vision, Technologies used, and Operating processes. It should be just a few pages long.

Requirements Documentation, of course, are helpful, even essential to have. Product requirements may be documented in many different forms, including for example the Product Backlog.

Acceptance Testing Documents are also helpful. Acceptance criteria is used by testers to determine whether the requirement has been met. Acceptance Testing Documents specify those acceptance criteria, and they help tell developers whether their work is complete, and whether it meets expectations of stakeholders.

Support Documentation is designed for those who will become responsible for supporting a project and a product once it's released. It may include Problem Escalation Procedures, a list of Important Contacts, and a Troubleshooting Guide.

User Documentation may include Training Manuals, User Manuals and Quick Reference Cards. It's designed to equip users to use a product and to find answers to their questions about it without having to call support staff.

So in Agile environments, there are still documents that are generated. However, the focus is on creating really effective communication, not documenting what doesn't need to be documented. And not spending too much time documenting things that you know will change later on in the project.

In summary, helpful documentation to create for an Agile project includes the Vision Statement, the Project Overview, Requirements Documentation, Acceptance Testing Documents, Support Documentation, and User Documentation.

## 12. Knowledge Check: Key Agile Concepts

* *identify characteristics of the Agile method*
* *distinguish between primary and secondary Agile values*
* *identify the five phases of the Agile project management model*
* *identify some of the methodologies that can be used for Agile project management*
* *recognize the four Scrum inspect and adapt events*
* *identify the five ADAPT steps required to transition to Agile*
* *identify the recommended components of a business case*
* *recognize the elements of a project charter*
* *identify the contract types suitable for Agile projects*
* *identify helpful types of documentation for Agile projects*

### **Question 1: Multiple Choice**

**What are five characteristics of the Agile method?**

#### **Options:**

1.

Agile is iterative

2.

Agile focuses on highest-value items first

3.

Agile identifies issues earlier, and changes are easier to implement

4.

Agile allows for feedback to be obtained early and often

5.

Agile is non-repetitive

6.

Agile focuses on the easiest requirements first

7.

Agile is incremental

#### **Answer**

1.

Agile is iterative

2.

Agile focuses on highest-value items first

3.

Agile identifies issues earlier, and changes are easier to implement

4.

Agile allows for feedback to be obtained early and often

7.

Agile is incremental

#### **Feedback:**

*Option 1:*

*That option is correct. The Agile methodology is iterative and processes and phases may be repeated as necessary for each deliverable as required.*

*Option 2:*

*This option is correct. Agile best practice is to focus on the highest-value requirements first to ensure they are completed and developed as required for the project.*

*Option 3:*

*This option is correct. In Agile projects, issues tend to surface early in the process, making it easier to address them as required.*

*Option 4:*

*This option is correct. A key characteristic of the Agile method is constant feedback as project work is completed, allowing for any necessary tweaks to be implemented so the final product meets requirements.*

*Option 5:*

*This option is incorrect. In an Agile environment, project work and phases may be repeated as many times as necessary to ensure the project meets requirements.*

*Option 6:*

*This option in incorrect. In an Agile project, the highest-value requirements are addressed first, rather than on requirements that might be fastest to complete.*

*Option 7:*

*That option is correct. Agile is incremental in nature, meaning work done in one phase is built upon in the next.*

### **Question 2: Multiple Choice**

**What are the four primary values identified in the Agile Manifesto?**

#### **Options:**

1.

Individuals and interactions over processes and tools

2.

Working software over comprehensive documentation

3.

Customer collaboration over contract negotiation

4.

Responding to change over following a plan

5.

Planning over flexibility

6.

Master processes over individual interactions

#### **Answer**

1.

Individuals and interactions over processes and tools

2.

Working software over comprehensive documentation

3.

Customer collaboration over contract negotiation

4.

Responding to change over following a plan

#### **Feedback:**

*Option 1:*

*This option is correct. One Agile value is to value individuals and interactions over processes and tools. Sometimes we mistakenly value certain mechanisms more than the individuals and interactions that create success when we should be thinking more about fostering greater communication.*

*Option 2:*

*This option is correct. Working software should be the measure of progress. The ability to start a project and quickly prototype something, or come up with something that works, is more valuable than working farther forward and documenting everything comprehensively.*

*Option 3:*

*This option is correct. In an Agile environment, it's understood that change is constant. So, the focus should be on collaboration with the customer to make sure that you understand the impact of the change, the need for the change, and the value that comes out of the change, instead of trying to negotiate whether or not to make changes.*

*Option 4:*

*This option is correct. You need to be able to respond to change and have a system that allows you to do so dynamically versus following a linear defined plan where you try to define everything upfront. Things are typically going to change over the course of a project.*

*Option 5:*

*This option is incorrect. In fact, the opposite is true. The strength of the Agile method is the ability to be flexible and adapt to changing needs and circumstances as they arise.*

*Option 6:*

*This option is incorrect. The Agile method values the individual and interactions over processes and tools. If a process is ultimately hindering creativity and innovation, it must be rethought or replaced.*

### **Question 3: Multiple Choice**

**What are the five phases of the Agile project management model?**

#### **Options:**

1.

Exploring

2.

Adapting

3.

Closing

4.

Monitoring

5.

Planning

6.

Speculating

7.

Envisioning

#### **Answer**

1.

Exploring

2.

Adapting

3.

Closing

6.

Speculating

7.

Envisioning

#### **Feedback:**

*Option 1:*

*This option is correct. Exploring is the third phase of the Agile project management model.*

*Option 2:*

*This option is correct. Adapting is the fourth phase of the Agile project management model.*

*Option 3:*

*This option is correct. Closing is the fifth and final phase of the Agile project management model.*

*Option 4:*

*This is an incorrect option. Monitoring is not one of the phases of the Agile project management model, which includes envisioning, speculating, exploring, adapting, and closing.*

*Option 5:*

*This option is incorrect. Planning in any project is important, but it is not one of the five phases of the Agile project management model, which includes envisioning, speculating, exploring, adapting, and closing.*

*Option 6:*

*This is a correct option. Speculating is the second phase of the Agile project management model.*

*Option 7:*

*This option is correct. Envisioning is the first phase of the Agile project management model.*

### **Question 4: Multiple Choice**

**Documentation is a secondary priority for Agile projects, but still valuable. What are the five types of documentation best suited for Agile environments?**

#### **Options:**

1.

Support and user documentation

2.

Acceptance testing documents

3.

Requirements documentation

4.

Detailed timeline plans

5.

Historical performance data

6.

Project overview

7.

Vision statement

#### **Answer**

1.

Support and user documentation

2.

Acceptance testing documents

3.

Requirements documentation

6.

Project overview

7.

Vision statement

#### **Feedback:**

*Option 1:*

*This is a correct option. Documents that help and support project work are important so work can be completed efficiently.*

*Option 2:*

*This is a correct option. It's important to document how project deliverables will be measured, and how they will be tested, to ensure the project meets the expected outcomes.*

*Option 3:*

*This option is correct. Requirements documentation is necessary so project work focuses on what the final product should look like, how it will work, and what it will do.*

*Option 4:*

*This is an incorrect option. Having a project timeline is important, but in an Agile environment, highly-detailed timeline plans are considered a hindrance rather than an effective aid.*

*Option 5:*

*This option is incorrect. Agile projects focus on completing tasks that contribute to the final project deliverable and addressing issues as they are identified, rather than documenting past results.*

*Option 6:*

*This is a correct option. While highly-detailed documentation is not useful in an Agile environment, having an effective project overview helps focus project work and priorities.*

*Option 7:*

*This option is correct. The vision statement is a valuable type of documentation as it guides the project work and outlines what the project is trying to achieve.*

## Course HTML Resources

| • | Glossary: Agile Principles and Methodologies |
| --- | --- |
|  |  |

| Ability | The third ADAPT step, where the actual ability for an organization to support an Agile approach is assessed. Factors that can impede change include issues with collocation of teams, engineering structure, or just not currently having people with the right skillsets to adopt Agile. |
| --- | --- |
| ADAPT steps | The five necessary requirements for successfully transitioning to Agile: Awareness, Desire, Ability, Promote, and Transfer. |
| adapting | Phase 4 of the Agile project management model, where changes or modifications to the plan take place as a result of what was learned in earlier phases. Changes may include reordering priorities or altering methodology to optimize teamwork. |
| Agile project management model | A five phase model that is cyclical, rather than linear. The phases are envisioning, speculating, exploring, adapting, and closing. |
| Awareness | The first ADAPT step, where an organization seeks to become aware of its problems or goals and how Agile methodologies can help to achieve goals or fix issues. |
| closing | Phase 5 of the Agile project management model, where a specific iteration, or the whole project, ends. |
| daily scrum | An opportunity to inspect and adapt as a team via a daily stand-up meeting, typically 15 minutes maximum, in which team members discuss what they've done, what they are going to work on, and any roadblocks or impediments to their work. |
| Desire | The second ADAPT step, where there is clear determination from the organization, or from enough people in the organization, to make a change and shift toward Agile methodologies. |
| envisioning | Phase 1 of the Agile project management model, where initial consideration is given to determine the project or product, business value, and vision for the overall project or produc |
| exploring | Phase 3 of the Agile project management model, where iterations of learning take place as a product is developed, tested, reviewed, and reworked in order to fulfill the vision created in the envisioning phase |
| Promote | The fourth ADAPT step, where a concerted effort is made to advocate for Agile across the organization, including the benefits that the organization can expect to gain. |
| scrum | A lightweight framework for Agile project management that utilizes an iterative, incremental approach which allows for predictability and better risk management. It consists of four events: sprint planning, daily scrum, sprint review, and sprint retrospective. |
| speculating | Phase 2 of the Agile project management model, where creative thinking starts with regard to how to implement different features or functionality in order to fulfill the vision created in the envisioning phase. |
| sprint planning | A meeting in which the team commits to a set of deliverables for the sprint. The product owner provides the team with feedback, which allows the opportunity to potentially reorder or reprioritize the requirements backlog. |
| sprint retrospective | A meeting in which the team assembles to discuss what went well, what didn't go well, and what action items they can take in order to improve as a team. |
| sprint review | A meeting held at the end of a sprint that allows the team to review and demo the product of their work to their stakeholders and to the product owner. |
| Transfer | The fifth ADAPT step, where the transition is made to begin implementing Agile methodologies or frameworks in the organization. |