I'm not a robot

Visual Paradigm Community - the Best #1 FREE Sequence Diagram Tool in the market UML modelling tool free for all sorts of non-commercial purpose. Supporting the 13 UML 2.x diagrams and ERD diagram. We are adopted by over 1 Million installations around the globe, and is still growing. Sequence Diagram is a model describing how groups of objects collaborate in some behavior over time which captures the behavior of a single use case. It often shows objects and the messages that are passed between these objects for the particular use case which can be used to model: A model describing how groups of objects collaborate in some behavior over time. The diagram captures the behavior of a single use case. It shows objects and the messages that are passed between these objects for the particular use case. Sequence Diagrams are interaction diagrams that detail how operations are carried out. Interaction diagrams model important runtime interactions between the parts that make up the system Model high-level interaction between active objects in a system Model the interaction between object instances within a collaboration that realizes a use case Model the interaction between objects within a collaboration that realizes an operation Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction Capture the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams) Capture high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams) Sequence Diagram is an interaction diagram that details how operations are carried out -- what messages are sent and when. Sequence diagrams are organized according to time. The time progresses as you go down the page. The objects involved in the operation are listed from left to right according to when they take part in the message sequence. Below is a sequence diagram for making a hotel reservation. The object initiating the sequence of messages is a Reservation window. Note That: Class and object diagrams are static model views. Interaction diagrams are dynamic. They describe how objects collaborate. A scenario is one path or flow through a use case that describes a sequence of events that occurs during one particular execution of a system. Sequence diagrams often be used to assist for elaborating use cases by emphasizing message exchange. The various scenarios of a use case can be depicted by a sequence diagram. The representation is restricted to the message exchange within each business use case. Generally, the level of detail for these sequence diagrams is higher than for sequence diagrams spanning use cases. A use case is a collection of interactions between external actors and a system. In UML, a use case is "the specification of a sequence of actions, including variants, that a system (or entity) can perform, interacting with actors of the system." Typically each use case includes a primary scenario (or main course of events) and zero or more secondary scenarios that are alternative courses of events to the primary scenario User requirements are captured as use cases that are refined into scenarios. A use case is a collection of interactions between external actors and a system. In UML, a use case is: "the specification of a sequence of actions, including variants, that a system (or entity) can perform, interacting with actors of the system." A scenario is one path or flow through a use case that describes a sequence of events that occurs during one particular execution of a system which is often represented by a sequence diagram. Sequence diagrams can be somewhat close to the code level, so why not just code up that algorithm rather than drawing it as a sequence diagram? A good sequence diagram is still a bit above the level of the real code Sequence diagrams are language neutral Non-coders can do sequence diagrams Easier to do sequence diagrams as a team Can be used for testing and/or UX Wireframing End-to-End Enterprise Architecture tool suite that supports TOGAF, ArchiMate, PMBOK process map and more Read More ❯ Agile development tools like user story map and sprint, Customer Journey Map and a wide range of project management diagrams Read More ❯ Essential development tools for modeling (UML, BPMN, ERD, DFD, etc), wireframing, code and DB engineering tools, etc Read More ❯ An award-winning modeling software that features UML, BPMN, ERD, DFD and more. Visual Paradigm Modeler makes modeling easy and fast. Read More ❯ Who's using this great UML software? Visual Paradigm are trusted by over 230,000 people in companies ranging from small business to Fortune 500 companies, universities and government units. Capture inspiration, stimulate creativity, and allow ideas to grow and flow freely. What you remember, what you see, what you think, and what you understand can go to infinity and beyond, and appear vividly on canvas. GitMind helps to complete the connection, flow, and co-creation of ideas, and accumulate & refine valuable ideas, empowering all to create their own metaverse of ideas. UML sequence diagram is particularly helpful in identifying the requirements of a system including all the functionalities. It’s easy to create sequence diagram as it only consists of basic shapes such as rectangles, lines, and arrows. You can work with it with simple tools like Word. There are also online tools to help you get this job done or you can use advanced solutions like Visio. Find out more useful methods by reading through this article.7 Practical Methods to Make Sequence DiagramCreate Sequence Diagram OnlineOne of the most popular solutions to create sequence diagram online is GitMind. The tool is surprisingly simple with lots of amazing features everyone may enjoy using. It comes with free and downloadable templates for easy diagramming. Also, it has an intuitive editor that grants you access to its shapes library for different types of UML diagrams. Moreover, you can export maps to image files or convert them into PDF files. Follow these simple steps to make a sequence diagram online.Go to GitMind and click “New Flowchart” from the templates library. From the editor, open the shapes library manager and choose UML shapes folders.Drag the shapes and components like the lifelines, objects, activation bar, arrows, and insert labels.Now export the map to your desired format by clicking the “Export” button at the top interface.Create Sequence Diagram in VisioVisio is another program that can help you create sequence diagram. This tool allows you to work offline via desktop app or online using the web version. With it, you can create, edit, view, and share diagrams without the need to install any extra app. What’s more, it has a great app integration with Microsoft software such as Excel and Word that will help you streamline processes or workflow further. Here are the steps on how to make sequence diagram using Visio.Open a new document and look for the search box then type UML sequence. Next, choose between Metric Units or US Units and click “Create.”It should load the diagram and the collection of shapes on the left side interface.Drag the elements for the sequence diagram from the shapes window to the page and double click on the text labels to rename.Create Sequence Diagram in ConcfluenceConfluence is a collaborative tool to work with your team in building and organizing projects in one place remotely wherever your location is. You can infuse plug-in like PlantUML, Gliffy, or ZenUML which you can use to use to learn how to make sequence diagrams. With Gliffy for Confluence, you can model numerous unified modeling language diagrams online. It also has a shape manager that gives you access to dynamic shapes and icons that suit business, software engineering, and project designs. See the instructions below and learn how to make a sequence diagram.Launch Gliffy on your browser and add shapes from the left-side panel.Organize the arrangement of the shapes and put labels to each process or event on the diagram.Finally, apply diagram themes to modify the appearance of the diagram then export or publish it.Create Sequence Diagram in PowerPointThe tool is primarily presentation software but is also a helpful method when you want to design a simple sequence diagram. You can create a sequence diagram with a PowerPoint shapes library. This method, however, requires you individually add and arrange the shapes to come up with a sequence diagram. Suppose you want to create sequence diagram in PowerPoint, here are steps you can follow.Open a blank document and go to the “Insert” tab then add the necessary shapes from the “Shapes” drop-down list.Identify the processes of the system you want to visualize and add them to the diagram. Just insert arrows and inserting labels with a text box.Customize your diagram from the “Format” tab where you can edit the shapes color, text color, font, and so on.Create Sequence Diagram in WordWord is another Microsoft product that you can use to create sequence diagram. In the same way, this lets you make a sequence diagram from scratch with the help of the shape library. Though this method gives your more flexibility in the creation process since you get to arrange them according to your desired structure. Also, you can add all the necessary shapes and customize every element.Open a new file and navigate to the “Insert” tab. From the “Shapes” menu, add all shapes that represent the objects, lifelines, messages, and activation bar.Once every element is inserted, label the messages and returning messages to clearly visualize the processes of the system.After you create a sequence diagram structure, you can add your personal touch to it by customizing its properties.Create Sequence Diagram in StarUMLStarUML is a diagramming tool that supports almost all platforms including Windows, Mac, and Linux. The good thing about this app is there are no big changes when it comes to its interface no matter the platform you are using. It comes with an extensive library of shapes for sequence diagrams, use case diagrams, class diagrams, flowcharts, and many more. You can also do specific customization to the elements. Allowing you to personalize the font, alignment, line style, and other properties. To know how to create sequence diagram, refer to the steps below.Install StarUML on your PC and launch it. Right-click on the “Untitled” folder and select “Add Diagram.” Choose “Sequence Diagram” from the option.A collection of dedicated elements will appear at the bottom left-hand side of the interface. Drag the notations you need and double click on the shape to add arrows, edit the text, add another lifeline, and more.For the final touch, adjust the properties of the elements from the “Styles” option. Once done, go to “File”, click “Save” and select a folder.Create Sequence Diagram in Draw.ioDraw.io does not only allow you to create sequence diagram but also an entity-relationship diagram, wireframe, network diagram, and mockups. Moreover, you can import your own set of shapes to work on a variety of graphical presentations or opt in using the pre-loaded shapes from its shapes manager. Aside from that, Draw.io has a built collaborative editing wherein teams can build and edit diagrams together virtually. Find out how this tool works from the step-by-step guide.Open a browser and launch Draw.io. Select between “Create New Diagram” and “Open Existing Diagram” options.A dialogue box containing templates will appear. Look for UML and select the sequence diagram illustration and click “Create” from the lower-left corner.Adjust the placement of the objects or messages on how you would according to the system you are visualizing. Once done, you can save the diagram on the cloud or share it by permitting access to another Google user.A sequence diagram is useful for businesses, software engineering and other process analyzing assignments. You can create sequence diagram with pen and paper but a software program designed for this purpose will save more time. Hence, the tools above along with their step-by-step tutorials should help you out. Skip to main contentLog inStart diagrammingStart diagrammingLog inStart diagrammingStart diagrammingTemplates homeWhether you’re working on a new software function or documenting a flow that works, FigJam’s sequence diagram tools can help you see how every component of your process fits together.Design the most effective processes for your systems (on the most engaging template for your team) with an interaction diagram.Start diagrammingGain a clear understanding of each use case, document detailed operations, and determine ways to improve processes already in motion.Avoid hiccups: Through feedback via cursor chats, emotes, and more, find and correct potential problems during the planning phase.Invite input: Interface with other developers or another participant to determine the best course of action for each use case.Get a birds’ eye view: Design a diagram that makes each step in a system’s process instantly intuitive.FigJamInteractions you won’t object toDevelop adaptable sequence diagrams with your team. In conjunction with your sequence diagram example, use Community-built tools and widgets including stamps, stickers, cursor chats, and audio chats to fold in feedback, discover new pathways, and keep the focus on collaboration—in your system and your team.A sequence diagram is a visual tool often that represents the interactions between different objects in a system in—you guessed it—a sequential order.In software engineering, a sequence diagram can be used to demonstrate how a function is carried out on a platform by breaking down when a certain fragment, objects or components interact. Sequence diagrams can also serve as useful frameworks for documenting workflows and operations and analyzing each interaction.Sequence diagrams visualize each step of an interaction or a function. Software developers often turn to sequence diagrams to map out the function’s sequence of events to spot any potential issues before they even begin hacking.Sequence diagrams can also make existing systems faster and more effective.Let’s say your customers are having issues purchasing specific movie seats using your online platform. By using sequence diagram examples, you can gain insight into the many steps the ticket buyer and computer server must complete before you can get butts in seats. From there, you can streamline the sequence of events—and your users’ experience.You can read a sequence diagram by beginning at the top of the diagram, where you’ll find entities—actors (people) or objects—in labeled rectangle boxes. Each object and interaction helps make sense of the world of software engineering in a sequence diagram example. Below each box, you’ll find a dotted “lifeline” which represents the different entities in the system, as well as arrows that will move you through the steps in the sequence.Wondering how to make sequence diagrams? With FigJam’s sequence diagram online tool, you can unpack your processes with ease. Just download our example of sequence diagram template to get started for free.Explore even more templates, widgets, and plugins—all built by the Figma community. This sequence diagram tutorial is to help you understand sequence diagrams better; to explain everything you need to know, from how to draw a sequence diagram to the common mistakes you should avoid when drawing one.There are 3 types of Interaction diagrams; Sequence diagrams, communication diagrams, and timing diagrams. These diagrams are used to illustrate interactions between parts within a system. Among the three, sequence diagrams are preferred by both developers and readers alike for their simplicity.In this sequence diagram tutorial you will learn about;What is a Sequence Diagram?Sequence diagrams, commonly used by developers, model the interactions between objects in a single use case. They illustrate how the different parts of a system interact with each other to carry out a function, and the order in which the interactions occur when a particular use case is executed.In simpler words, a sequence diagram shows how different parts of a system work in a ‘sequence’ to get something done.Sequence diagrams are commonly used in software development to illustrate the behavior of a system or to help developers design and understand complex systems. They can be used to model both simple and complex interactions between objects, making them a useful tool for software architects, designers, and developers.Sequence Diagram NotationsA sequence diagram is structured in such a way that it represents a timeline that begins at the top and descends gradually to mark the sequence of interactions. Each object has a column and the messages exchanged between them are represented by arrows.A Quick Overview of the Various Parts of a Sequence DiagramLifeline NotationA sequence diagram is made up of several of these lifeline notations that should be arranged horizontally across the top of the diagram. No two lifeline notations should overlap each other. They represent the different objects or parts that interact with each other in the system during the sequence.A lifeline notation with an actor element symbol is used when the particular sequence diagram is owned by a use case.A lifeline with an entity element represents system data. For example, in a customer service application, the Customer entity would manage all data related to a customer.A lifeline with a boundary element indicates a system boundary/ software element in a system; for example, user interface screens, database gateways or menus that users interact with, are boundaries.And a lifeline with a control element indicates a controlling entity or manager. It organizes and schedules the interactions between the boundaries and entities and serves as the mediator between them.Activation BarsThe activation bar is the box placed on the lifeline. It is used to indicate that an object is active (or instantiated) during an interaction between two objects. The length of the rectangle indicates the duration of the objects staying active.In a sequence diagram, an interaction between two objects occurs when one object sends a message to another. The use of the activation bar on the lifelines of the Message Caller (the object that sends the message) and the Message Receiver (the object that receives the message) indicates that both are active/ are instantiated during the exchange of the message.Message ArrowsAn arrow from the Message Caller to the Message Receiver specifies a message in a sequence diagram. A message can flow in any direction; from left to right, right to left, or back to the Message Caller itself. While you can describe the message being sent from one object to the other on the arrow, with different arrowheads you can indicate the type of message being sent or received.The message arrow comes with a description, which is known as a message signature, on it. The format for this message signature is below. All parts except the message_name are optional.attribute = message_name (arguments): return_type As shown in the activation bars example, a synchronous message is used when the sender waits for the receiver to process the message and return before carrying on with another message. The arrowhead used to indicate this type of message is a solid one, like the one below.An asynchronous message is used when the message caller does not wait for the receiver to process the message and return before sending other messages to other objects within the system. The arrowhead used to show this type of message is a line arrow as shown in the example below.A return message is used to indicate that the message receiver is done processing the message and is returning control over to the message caller. Return messages are optional notation pieces, for an activation bar that is triggered by a synchronous message always implies a return message.Tip: You can avoid cluttering up your diagrams by minimizing the use of return messages since the return value can be specified in the initial message arrow itself.Participant creation messageObjects do not necessarily live for the entire duration of the sequence of events. Objects or participants can be created according to the message that is being sent.The dropped participant box notation can be used when you need to show that the particular participant did not exist until the create call was sent. If the created participant does something immediately after its creation, you should add an activation box right below the participant box.Participant destruction messageLikewise, participants when no longer needed can also be deleted from a sequence diagram. This is done by adding an ‘X’ at the end of the lifeline of the said participant.When an object sends a message to itself, it is called a reflexive message. It is indicated with a message arrow that starts and ends at the same lifeline as shown in the example below.UML diagrams generally permit the annotation of comments in all UML diagram types. The comment object is a rectangle with a folded-over corner as shown below. The comment can be linked to the related object with a dashed line.Note: View Sequence Diagram Best Practices to learn about sequence fragments.Sequence Diagram Best PracticesManage complex interactions with sequence fragmentsA sequence fragment is represented as a box that frames a section of interactions between objects (as shown in the examples below) in a sequence diagram.It is used to show complex interactions such as alternative flows and loops in a more structured way. On the top left corner of the fragment sits an operator. This – the fragment operator – specifies what sort of a fragment it is.AlternativesThe alternative combination fragment is used when a choice needs to be made between two or more message sequences. It models the “if then else” logic.The alternative fragment is represented by a large rectangle or a frame; it is specified by mentioning ‘alt’ inside the frame’s name box (a.k.a. fragment operator).To show two or more alternatives, the larger rectangle is then divided into what is called interaction operands using a dashed line, as shown in the sequence diagram example above. Each operand has a guard to test against and it is placed at the top left corner of the operand.OptionsThe option combination fragment is used to indicate a sequence that will only occur under a certain condition, otherwise, the sequence won’t occur. It models the “if then” statement.Similar to the alternative fragment, the option fragment is also represented with a rectangular frame where ‘opt’ is placed inside the name box.Unlike the alternative fragment, an option fragment is not divided into two or more operands. Option’s guard is placed at the top left corner.(Find an example sequence diagram with an option fragment in the Sequence Diagram Templates and Examples section).LoopsLoop fragment is used to represent a repetitive sequence. Place the words ‘loop’ in the name box and the guard condition near the top left corner of the frame.In addition to the Boolean test, the guard in a loop fragment can have two other special conditions tested against. These are minimum iterations (written as minint = [the number] and maximum iterations (written as maxint = [the number]).If it is a minimum iterations guard, the loop must execute not less than the number mentioned, and if it is a maximum iterations guard, the loop mustn’t execute more than the number indicated.(Find an example of a loop fragment below in the sequence diagram templates and example section)Reference FragmentYou can use the ref fragment to manage the size of large sequence diagrams. It allows you to reuse part of one sequence diagram in another, or in other words, you can reference part of a diagram in another diagram using the ref fragment.To specify the reference fragment, you have to mention ‘ref’ in the name box of the frame and the name of the sequence diagram that is being referred to inside the frame.For more sequence fragments refer to Beyond the Basics of Sequence Diagrams: Part 1,Part 2 and Part 3.Draw smaller sequence diagrams that capture the essence of the use caseInstead of cluttering your sequence diagram with several objects and groups of messages that will confuse the reader, draw a few smaller sequence diagrams that aptly explain what your system does. Make sure that the diagram fits on a single page and leaves space for explanatory notes too.Also instead of drawing dozens of sequence diagrams, find out what is common among the scenarios and focus on that. And if the code is expressive and can stand on its own, there’s no need to draw a sequence diagram in the first place.How to Draw a Sequence DiagramA sequence diagram represents the scenario or flow of events in one single use case. The message flow of the sequence diagram is based on the narrative of the particular use case.Then, before you start drawing the sequence diagram or decide what interactions should be included in it, you need to draw the use case diagram and ready a comprehensive description of what the particular use case does.From the above use case diagram example of ‘Create New Online Library Account’, we will focus on the use case named ‘Create New User Account’ to draw our sequence diagram example.Before drawing the sequence diagram, it’s necessary to identify the objects or actors that would be involved in creating a new user account. These would be;LibrarianOnline Library Management systemUser credentials databaseEmail systemOnce you identify the objects, it is then important to write a detailed description of what the use case does. From this description, you can easily figure out the interactions (that should go in the sequence diagram) that would occur between the objects above, once the use case is executed.Here are the steps that occur in the use case named ‘Create New Library User Account’.The librarian request the system to create a new online library accountThe librarian then selects the library user account typeThe librarian enters the user’s detailsThe user’s details are checked using the user Credentials DatabaseThe new library user account is createdA summary of the new account’s details is then emailed to the userFrom each of these steps, you can easily specify what messages should be exchanged between the objects in the sequence diagram. Once it’s clear, you can go ahead and start drawing the sequence diagram.The sequence diagram below shows how the objects in the online library management system interact with each other to perform the function ‘Create New Library User Account’.Sequence Diagram Common MistakesWhen drawing sequence diagrams, designers tend to make these common mistakes. By avoiding these mistakes you can ensure the quality of your diagram.Adding too much detail. This clutters up the diagram and makes it difficult to read.Obsolete and out-of-date sequence diagrams that are irrelevant when compared to the interfaces, actual architectures, etc. of the system. Don’t forget to replace them or modify them.Leaving no blank space between the use case text and the message arrow; this makes it difficult for anyone to read the diagram.Not considering the origins of message arrows carefully.See these common mistakes explained in detail in Sequence Diagram Guide: Common Mistakes to Avoid When Drawing Sequence Diagrams.Sequence Diagram Templates and ExamplesFollowing are a few sequence diagram examples and templates that are drawn using Creately. Create sequence diagrams online using Creately’s online tool. Click on the template to open it in the editor.Sequence Diagram of an Online Exam SystemOnline Examination - Sequence Diagram Template (Click on the template to edit it online)Sequence Diagram Example of a School Management SystemSchool Management System - Sequence Diagram Template (Click on the template to edit it online)Example of an Option Combination FragmentExample of an option fragment (Click on the template to edit it online)Example of a Loop SequenceLoops - Sequence Diagram Example (Click on the template to edit it online)Sequence Diagram Example of a Card GameCard Game - Sequence Diagram Template (Click on the template to edit it online)Sequence Diagram Example of a Balance LookupBalance Lookup - Sequence Diagram Example (Click on the template to edit it online)Sequence Diagram Example of an Online Movie Ticket Booking SystemOnline Movie Ticket Booking - Sequence Diagram Template (Click on the template to edit it online)Here are some more sequence diagram templates and examples that you can edit right away.Feedback on the Sequence Diagram GuideThis sequence diagram tutorial covers everything you need to know on sequence diagrams and drawing them. If you have any suggestions or questions regarding the sequence diagram tutorial, feel free to leave a comment.More Diagram TutorialsSequence diagrams provide a simplified view of complex system interactions, making it easier to understand the system’s behavior.Sequence diagrams provide a common language for developers, designers, and other stakeholders to discuss the system’s behavior which contributes to improving communication.Sequence diagrams can help to identify errors and issues in the system making it easier to identify and fix problems before they become more serious.Sequence diagrams can be used to design new systems, allowing developers to test different scenarios and identify potential issues before they start coding. This helps save time and resources by identifying problems early in the development process.UML stands for Unified Modeling Language, and it is a standardized visual language for modeling software systems. UML diagrams are graphical representations of different aspects of a software system, such as its structure, behavior, and interactions between components. They are typically used to communicate design ideas and requirements to stakeholders, to understand and analyze complex software systems, and to guide the implementation of software systems.To learn more about different types of UML diagrams, refer to Creately’s guide on UML Diagram Types.In sequence diagrams, lifelines are depicted as vertical lines that represent the objects or actors participating in the interaction. In contrast, communication diagrams use a horizontal line to represent each object or actor, and arrows are used to represent messages exchanged between them.In sequence diagrams, messages are represented by arrows that indicate the order in which they are sent and received. The arrow typically points downwards from the sender to the receiver. In communication diagrams, messages are represented by arrows that connect the lifelines and indicate the direction of the message flow. The arrow typically points from the sender to the receiver.Sequence diagrams place a greater emphasis on the timing and order of messages exchanged between objects. Therefore, they are more suitable for modeling interactions that require a detailed understanding of the timing and order of events. In contrast, communication diagrams focus on the relationships between objects and the messages exchanged between them, making them more suitable for modeling interactions that require a high-level view of the system’s structure. Enjoy the new generation of software development with contemporary DevOps tools. UX Design Visualize screen flow and layout with wireframe, storyboard, and prototype tool. Learn more An online editor that provides content creators with charts, data widgets, and maps to visualize data while providing information, tables, and spreadsheet tools to increase productivity. Check Visual Paradigm online Learn more Work simultaneously and collaboratively on the same project. Keep your work securely in the cloud workspace. Access it anytime, anywhere. Learn more Want to learn more? You'll find simple and state-of-the-art tools in Visual Paradigm. The go-to solution for leading product teams worldwide. Join the 320,000+ professionals and organizations who trust us! From small businesses to Fortune 500 companies, universities, and government sectors, our proven solutions have made us the go-to choice for enterprise success. The world's leading institutions use Visual Paradigm to prepare the next generation of IT developers with the specialized skills needed for the workspace. Join Us Now! We use cookies to offer you a better experience. By visiting our website, you agree to the use of cookies as described in our Cookie Policy. Welcome to the comprehensive tutorial on creating Sequence Diagrams in UML. Sequence diagrams are a vital part of UML, allowing you to visually represent interactions between objects in a system. This tutorial will guide you through the notations, elements, and steps involved in drawing effective sequence diagrams. Whether you’re a beginner or looking to enhance your UML skills, this guide is designed to provide a clear understanding of sequence diagrams and how to use them in your software modeling endeavors. What is a Sequence Diagram in UML? UML Sequence diagrams are a powerful tool for capturing and visualizing interactions between objects in a system. They are particularly useful in transitioning from high-level requirements, such as use cases, to a more formal level of system design. Sequence diagrams illustrate the chronological order of interactions, depicting lifelines and messages exchanged between objects. When to Use Sequence Diagrams? Model high-level interactions between active objects in a system. Model interactions within a collaboration that realizes a use case. Model interactions within a collaboration that realizes an operation. Capture either generic interactions (showing all possible paths) or specific instances of an interaction (showing just one path). Sequence Diagram Notations Lifeline A lifeline represents an individual participant in the interaction. Actor An actor is a role played by an entity interacting with the subject. It can represent human users, external hardware, or other subjects. Activation An activation, represented by a thin rectangle on a lifeline, signifies the period during which an element is performing an operation. Messages Call Message A call message defines communication between lifelines, representing the invocation of an operation on the target lifeline. Return Message A return message represents the passing of information back to the caller of a corresponding former message. Self Message A self message denotes communication within the same lifeline, representing the invocation of a message on itself. Recursive Message A recursive message is similar to a self message but points to an activation on top of the current one. Create Message A create message signifies the instantiation of a target lifeline. Destroy Message A destroy message represents the request to destroy the lifecycle of the target lifeline. Duration Message A duration message shows the time distance between two time instants for a message invocation. Note A note or comment provides the ability to attach remarks to elements but carries no semantic force. Drawing a Sequence Diagram: Step-by-Step Guide Identify Participants: Determine the objects participating in the collaboration or use case scenario. Scenario Analysis: If derived from a scenario, select the normal scenarios first. Identify the primary actor(s) activating the use case. Message Flow: Consider the initiating point of the scenario. Define the system’s response to the actor’s message and what needs handling before the return message. Object and Operation Identification: Identify candidate objects and operations based on the scenario. Use this information to incrementally derive the class diagram. Repeat Scenario Points: Iterate through each point of the scenario until completion. Alternative Scenarios: Draw corresponding sequence diagrams for exception or alternative scenarios. Sequence Diagram Examples Sequence in MVC Framework Pattern In the following sequence diagram example, we observe the interactions between a user and a set of participating objects. The diagram comprises four primary components: the user, represented as the actor, the boundary object named ‘interface,’ the controller object identified as ‘mainController,’ and two entity objects named ‘routes’ and ‘route.’ Hospital Bed Allocation Recursive Message Example Get Started: Draw Your Sequence Diagram Ready to draw your Sequence Diagram? Utilize online tools like Visual Paradigm Online, offering free usage for non-commercial purposes. Draw Your Sequence Diagram Now Remember, mastering sequence diagrams takes practice, so dive in and enjoy the process of creating visual representations of complex interactions in your system! Tips and Tricks for Using Sequence Diagrams in UML Clearly Define Participants: Clearly identify and name the participants (objects or actors) involved in the interaction. This ensures clarity in understanding the roles played by each entity. Keep It Simple: Strive for simplicity. Avoid unnecessary details that can clutter the diagram. Focus on capturing essential interactions to convey the intended message. Use Descriptive Naming: Employ descriptive and meaningful names for lifelines, messages, and objects. This enhances the readability and comprehension of the sequence diagram. Group Related Messages: Group related messages together to represent a coherent flow of interactions. This helps in maintaining a logical and organized structure. Timing and Duration Constraints: Leverage timing and duration constraints to depict the temporal aspects of message exchanges. This is especially useful when illustrating delays or specific time intervals. Consider Alternative Scenarios: Explore alternative scenarios and exception paths. Sequence diagrams are not only for depicting the main flow but also for showcasing how the system behaves under different conditions. Use Interaction Fragments Wisely: Employ interaction fragments such as loops, alternatives, and options to capture complex behavior. These fragments add depth to the diagram and accommodate various scenarios. Limit Self Messages: While self messages are useful for depicting actions within the same lifeline, avoid overusing them. Excessive self messages can lead to a cluttered diagram. Pay Attention to Activation Bars: Ensure activation bars are appropriately aligned with the initiation and completion times of the corresponding lifeline. This accuracy aids in understanding the temporal flow of the interactions. Consistent Notation: Maintain consistency in notation throughout the diagram. This includes using the same symbols for messages, properly aligning activation bars, and adhering to UML conventions. Annotate with Notes: Use notes to provide additional context or explanations. This is especially helpful when certain aspects of the interaction need clarification. Validate with Stakeholders: Validate your sequence diagrams with stakeholders to ensure that the depicted interactions align with their understanding of the system. This helps in refining the diagrams for accuracy. Tool Proficiency: Familiarize yourself with UML modeling tools that support sequence diagrams. Proficiency in these tools can significantly enhance your productivity and the quality of the diagrams. Iterative Refinement: Sequence diagrams are often refined iteratively. Don’t hesitate to revisit and refine your diagrams as your understanding of the system evolves. Practice Regularly: Like any skill, creating effective sequence diagrams improves with practice. Regularly challenge yourself with diverse scenarios to hone your UML modeling skills. By incorporating these tips and tricks, you’ll be better equipped to create clear, concise, and accurate sequence diagrams that effectively communicate the dynamic interactions within your systems. In this tutorial, we explored the fundamental concepts of UML Sequence Diagrams, starting with an understanding of their purpose and usage in system design. We delved into the notations, including lifelines, actors, activations, and various message types. The tutorial provided a step-by-step guide on when and how to draw sequence diagrams, emphasizing scenario analysis, object identification, and the iterative process of capturing interactions. Additionally, we showcased practical examples, such as booking a seat and hospital bed allocation, to illustrate the application of sequence diagrams in real-world scenarios. The tutorial concluded by encouraging you to draw your sequence diagrams using tools like Visual Paradigm Online. Now equipped with the knowledge and tools, you’re ready to confidently create sequence diagrams, effectively capturing the dynamic interactions within your systems. Dive in, experiment, and enjoy the process of translating complex scenarios into visual representations! Share — copy and redistribute the material in any medium or format for any purpose, even commercially. Adapt — remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms. Attribution — You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation . No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material. title THIS IS A LARGE TITLEWITH A LINE BREAK participantspacing 14 Alice->Bob:info Alice New participants of type participant may be added by clicking the icon The following special participant types exist participant - box with participant name inside (the default) rparticipant - rounded box actor - specific icon boundary - specific icon control - specific icon database - specific icon entity - specific icon materialdesignicons - unicode code points (see right hand side in the gray bar after clicking an icon) fontawesome6solid - unicode code points (see top right corner after clicking an icon) fontawesome6regular - unicode code points (see top right corner after clicking an icon) fontawesome6brands - unicode code points (see top right corner after clicking an icon) image - png image data url, maximum data url size is 50KB, the rendered size is calculated based on maximum participant height, add padding in the top of the image to decrease rendered width and total size note: fontawesome5solid/regular/brands and fontawesome (v4) are now deprecated but will remain supported until at least 2025 A long displayed name can be written on form: participant "some verylong name" as Alice Edit a participant by double-clicking it Change the position of participant by clicking and dragging it to the right or left passed another participant Delete a participant (and all messages interactions with it) by clicking it and using the delete key participant Participant actor "**++Big andbold name" as actorMultiline #red participant "some longname with **//styling//**" as participantMultiline materialdesignicons F1FF escalator fontawesome6solid f48e "++**Syringe**++" as Syringe #red fontawesome6regular f0f8 Hospital #blue fontawesome6brands f3b6 Jenkins #green actor #green:0.5 Actor boundary #ff00ff:2 Boundary control :1 Control database #blue:1 Database #red entity :0.5 Entity participant :0 Participant participant :0 "++**Participant 2**++" as p2 image ... png The participants can be displayed in the bottom of the diagram by using the bottomparticipants keyword which renders all the participants in the bottom of the diagram Note: An alternative for displaying participants when scrolling in a large diagram is "View" -> "Participant Overlay On Scroll" bottomparticipants participant A participant "BBBBBBBB" as B materialdesignicons f14d note fontawesome6regular f0f8 Hospital fontawesome6brands f3b6 Jenkins actor Actor A->B:info A->note:info A->Hospital:info A->Jenkins:info A->Actor:info Messages are created by clicking and dragging in the diagram Hold Shift before clicking for dashed line Hold Ctrl (or Cmd for Mac) before clicking for open arrow Hold Shift+Ctrl (or Shift+Cmd for Mac) before clicking for open arrow with dashed line Edit the text of a message by double-clicking it Change the start and end participants of the message by clicking and dragging the start or end of the message Change position of the message by clicking and dragging the middle of the message Delete the message by clicking it and pressing the delete key New line is create using Alice-:4>Bob:Test12345 Alice-:2>>Bob:Test Alice-#00ff00:2>Bob:Test Alice--#red:4>Bob:Test AliceBob:Test AliceBob:Test AliceBob:Test AliceBob:Test AliceBob:Test Non-instantaneous messages are created by adding ([delay]) before the target participant, examples: A-(1)B: info A-->(5)B: info A->>(2)B: info Otherwise non-instantaneous messages behaves just like normal messages participant A participant B participant C A->(1)B:info A(1)(5)C:infoinfo activate B activate C B(1) Client->(5)Server:first sent message space -6 Client->Server:later message Incoming and Outgoing Messages are created by using the special participants [and], examples: Failure Messages are created by using x to denot the arrow head, examples: A-xB: info A--#redx(1)C: info A-#redxB:failure 1 C-xB:failure 2 Bx-B:failure 3 C(5)x--A:failure 4 Notes and boxes are created by right clicking in the diagram and selecting the wanted note / box entry from the menu Edit the text of a note or box by double clicking it Change the start and end participants of the "note or box over several participants" by clicking and dragging the start or end of the note or box Change position of the note or box by clicking and dragging the middle of the note or box Note: It is the bottom of the shapes that counts as the y position when dragging Delete the note or box by clicking it and pressing the delete key New line is create using note over A:note over onemultiple linesof text note over A,B:note over several note left of A:note left of note right of A:note right of box over A:box over one box over A,B:box over several box left of A:box left of box right of A:box right of abox over A:abox over one abox over A,B:abox over several abox left of A:abox left of abox right of A:abox right of rbox over A:rbox over one rbox over A,B:rbox over several rbox left of A:rbox left of rbox right of A:rbox right of aboxright over A,B:This is angular boxright aboxleft over A,B:This is angular boxleft aboxright over A:This is angular boxright aboxleft over B:This is angular boxleft aboxright right of A:This is angular boxright aboxright left of B:This is angular boxright aboxleft right of A:This is angular boxright aboxleft left of B:This is angular boxleft References are created by right clicking in the diagram selecting over which participants the reference should be from the menu Edit the text of a reference by double clicking it Change the start and end participants of the reference by clicking and dragging the start or end of the reference Change position of the reference by clicking and dragging the middle of the note or box Note: It is the bottom of the shapes that counts as the y position when dragging Delete the reference by clicking it and pressing the delete key New line is create using A->B:info ref over B,C:other interaction C->D:info Dividers are created by right clicking in the diagram and selecting the divider entry from the menu Edit the text of a divider by double clicking it Change position of the divider by clicking and dragging it Delete the divider by clicking it and pressing the delete key participant A participant B participant C participant D ==info== Create and destroy are at this point not part of the context menu, participants may be defined in the start of the diagram participantNameA->*participantNameB: message: Sends a message to participantNameB and creates participantNameB create participantName: Creates the participant without sending a message to it destroy participantName: Destroys the participant at the previous entry's y position destroyafter participantName: Destroys the participant at after a space and gives the destroy symbol its own space destroysilent participantName: Destroys the participant at the previous entry's y position without rendering the destroy symbol Click and drag on the entries to move them in y axis using the mouse participant A actor X participant B A->B:info B-->*C: note over C:do something B*X:message to X note over X:do something destroyafter X A->B:info A->B:info create C note over C: C created without message A A->>B:request AD:info Participant spacing allows control of spacing between the participants The participantspacing equal statement makes the spacing between all participants equal The participantspacing 50 statement makes the spacing between all participants at least 50 participantspacing equal participant A participant B participant C A->B:info info info Entry spacing allows control of spacing between the entries Click anywhere on the diagram and press the + or - key to change spacing between all entries Add entryspacing statements to change in different places of the diagram entryspacing 0.1 A->B:info A->B:info entryspacing 3 A->B:info entryspacing 1 A->B:info A->B:info Specifies the style of the life lines lifelinestyle #blue - make all life lines blue lifelinestyle participantName #red - make participant life line red lifelinestyle :4 - make all life line weights 4 lifelinestyle C #gray:1:solid - make participant life line gray, line weight 1, solid participant A participant B participant C participant D lifelinestyle #red:4:solid lifelinestyle B #black:1:dashed lifelinestyle C #gray:1:solid lifelinestyle D ::dashed A->B:info B->C:info C->D:info Large example including most of the basic elements Source too large to display