MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

1: Developing Business Central Extensions (part 3) â€“ Freddys Blog

Ã€ propos de ce cours: "Machine Design Part I" is the first course in an in-depth three course series of "Machine Design."The "Machine Design" Coursera series covers fundamental mechanical design topics, such as static and fatigue failure theories, the analysis of shafts, fasteners, and gears, and the design of mechanical systems such as gearboxes.

It is used in server, workstation, and high-end embedded scenarios think digital video recorders and routers, not cell phones. The Cell Broadband Engine is the up-and-coming architecture using the PowerPC instruction coupled with eight vector processors. The Sony PlayStation 3 will use the Cell, as well as numerous other vendors considering it for a wide variety of multimedia applications. The instruction set itself can operate in either a bit mode or a reduced bit mode. G4s and earlier are only bit. High level versus low level programming Most programming languages are fairly processor-independent. While they may have specific features that rely on certain processor abilities, they are more likely to be operating-system-specific than processor-specific. These high-level programming languages are built for the express purpose of providing distance between the programmer and the hardware architecture. This is for several reasons. While portability is one of them, probably more important is the ability to provide a friendlier model that is geared more towards how programmers think as opposed to how the chip is wired. This means that you have essentially the same view of the system that the hardware does. This has the potential to make assembly language programming more difficult because the programming model is geared towards making the hardware work instead of closely mirroring the problem domain. The benefits are that you can do system-level work easier and perform optimizations that are very processor-specific. The drawbacks are that you actually have to think on that level, you are tied to a specific processor line, and you often have to do a lot of extra work to get the problem domain accurately modeled. In high-level languages, there is a lot going on with every expression. You sometimes have to wonder just what is occurring under the hood. In assembly language programming, you can have a full grasp of exactly what the hardware is doing. You can step through the hardware-level changes every step of the way. Fundamentals of assembly language Before getting into the instruction set itself, the two keys to understanding assembly language are understanding the memory model and understanding the fetch-execute cycle. The memory model is very simple. Memory stores only one thing -- numbers with a limited range called a byte on most computers, this is a number between 0 and Each memory location is located using a sequential address. Think of a giant roomful of post-office boxes. Each box is numbered, and each box is the same size. This is the only thing that computers can store. Therefore, everything must ultimately be structured in terms of fixed-range numbers. Thankfully, most processors have the ability to combine multiple bytes together as one unit to handle larger numbers, and also numbers with different ranges such as floating-point numbers. However, how specific instructions treat a region of memory is irrelevant to the fact that every memory location is stored in the exact same manner. In addition to the memory lying in sequential addresses, processors also maintain a set of registers, which are temporary locations for holding data being manipulated or configuration switches. The fundamental process that controls processors is the fetch-execute cycle. Processors have a register known as the program counter, which holds the address of the next instruction to execute. The fetch-execute works in the following way: The program counter is read, and the instruction is read from the address listed there The program counter is updated to point to the next instruction The instruction is decoded All memory items needed to process the instruction are loaded The computation is processed The results are stored The reality of how this occurs is actually much more complicated, especially since the POWER5 processor can execute up to five instructions simultaneously. However, this suffices for a mental model. This means that all calculations are performed on registers, not main memory. Memory access is simply for loading data into registers and storing data from registers into memory. This is different from, say, the x86 architecture, in which nearly every instruction can operate on memory, registers, or both. The PowerPC has 32 general-purpose registers and 32 floating-point registers, which are each numbered as apposed to the x86, where the registers are named rather than numbered. It also has a few special-purpose registers for holding status information and return addresses. There are other special-purpose registers available to supervisor-level applications, but these are beyond the scope of this article. The general-purpose registers are 32 bits on bit architectures and 64 bits on bit architectures. This article series focuses on the bit architectures. Instructions in assembly language are very low-level -- they can only perform one or sometimes a few operations at a time. This may seem tedious at times. However, there are three important benefits. First, simply knowing assembly language will help you write better high-level code, since you will understand what is going on at the lower levels. Second, the fact that you have access to all of the minutia of assembly language means that you can optimize speed-critical loops beyond what your compiler can do. Compilers are fairly good at code optimization. However, knowing assembly language can help you understand both the optimizations that the compiler is making using the -S switch in gcc has the compiler spitting out assembly rather than object code , and help you spot the ones that it is missing. Third, you will have access to the full power of the PowerPC chip, much of which can actually make your code more compact than possible in higher-level languages. Here are some PowerPC instructions that are useful to beginners: In all of these instructions the registers are specified only by their number. For example, the instruction for loading the number 12 into register 5 is li 5, We know that 5 refers to a register and 12 refers to the number 12 because of the instruction format -- there is no other indicator. Each PowerPC instruction is 32 bits long. The first six bits determine the instruction and the remaining portions have different functions depending on the instruction. The fact that they are fixed-length allows the processor to process them more efficiently. However, the limitation to 32 bits can cause a few headaches, which we will encounter. The solutions to most of these headaches will be discussed in part 2. Many of the instructions above make use of the PowerPC extended mnemonics. This means that they are actually specializations of a more general instruction. For example, all of the conditional branches mentioned above are actually specializations of the bc branch conditional instruction. The CBIT is the bit of the condition register to test. The MODE has lots of interesting uses, but for simple uses you set it to 12 if you want to branch if the condition bit is set, and 4 if you want to branch if the condition bit is not set. Some of the important condition register bits are 8 for less than, 9 for greater than, and 10 for equal. Likewise li is a specialized form of addi and mr is a specialized form of or. These extended mnemonics help make PowerPC assembly language programs more readable and writable for simpler programs, while not removing the power available for more advanced programs and programmers. Type the following into a file named sum. The first step in building this program is to assemble it: The "-m64" switch tells the assembler that you are using the bit ABI as well as bit instructions. The generated object code is the machine-language form of the code, however, it cannot be run directly as-is. It needs to be linked it so that it is ready for the operating system to load it and run it. To link, do the following: To run the program, do: This will print out "3", which is the final result. Since assembly language code works very close to the level of the operating system itself, it is organized very closely to the object and executable files which it will produce. So, to understand the code, we first need to understand object files. Object and executable files are divided up into "sections". Each section is loaded into a different place in the address space when the program is executed. They have different protections and purposes. The main sections we will concern ourselves with are: We force the section to align at an 8-byte boundary. The next three data items generated are the procedure descriptor, which will be discussed in a later article. Now we can switch to actual program code. After this is where. The first set of instructions loads in the address of the first value not the value itself. Therefore, since the address can be up to 64 bits, we have to load it a piece at a time part 2 of this series will show how to avoid this. The following are used here: Bits of register 7 now contain bits of the address. Now bits of the address are in bits of register 7. Register 7 is now shifted left 32 bits, with bits cleared out, and the result is stored in register 7. Now bits of register 7 contain bits of the address we are loading. The next two instructions use "or immediate" and "or immediate shifted" instructions to load bits in a similar manner. In the next article we will cover the addressing modes that make this easier. Now, remember, this only loads in the address of the value we want to load. Now we want to load the value itself into a register. To do this, we will use register 7 to tell the processor what address we want to load the value from. This will be indicated by putting "7" in parentheses. The instruction ld 4, 0 7 loads the value at the address in register 7 into register 4 the zero means to add zero to that address.

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

2: In the beginning VS AT the beginning | WordReference Forums

Note: I want a detailed clear answers please. The Mexton Machines Company was founded in the 's on an old RAF airfield in the East Midlands of England.

History[edit] The definition of the Internet of things has evolved due to convergence of multiple technologies, real-time analytics , machine learning , commodity sensors, and embedded systems. The extensive set of applications for IoT devices [24] is often divided into consumer, commercial, industrial, and infrastructure spaces. A smart home or automated home could be based on a platform or hubs that control smart devices and appliances. These features can include sensors that monitor for medical emergencies such as falls or seizures. By , it is estimated that EIoT will account for 9. These health monitoring devices can range from blood pressure and heart rate monitors to advanced devices capable of monitoring specialized implants, such as pacemakers, Fitbit electronic wristbands, or advanced hearing aids. It can also adjust itself to ensure appropriate pressure and support is applied to the patient without the manual interaction of nurses. IoMT in the healthcare industry is now permitting doctors, patients and others involved i. This includes sensor-based solutions such as biosensors, wearables, connected health devices and mobile apps to track customer behaviour. This can lead to more accurate underwriting and new pricing models. The IoT can assist in the integration of communications, control, and information processing across various transportation systems. Application of the IoT extends to all aspects of transportation systems i. Dynamic interaction between these components of a transport system enables inter and intra vehicular communication, [56] smart traffic control , smart parking, electronic toll collection systems , logistic and fleet management , vehicle control , and safety and road assistance. If combined with Machine Learning then it also helps in reducing traffic accidents by introducing drowsiness alerts to drivers and providing self driven cars too. Building and home automation[edit] IoT devices can be used to monitor and control the mechanical, electrical and electronic systems used in various types of buildings e. In this context, three main areas are being covered in literature: Based on such a highly integrated smart cyberphysical space, it opens the door to create whole new business and market opportunities for manufacturing. Measurements, automated controls, plant optimization, health and safety management, and other functions are provided by a large number of networked sensors. IIoT in manufacturing could generate so much business value that it will eventually lead to the fourth industrial revolution, so the so-called Industry 4. It is estimated that in the future, successful companies will be able to increase their revenue through Internet of things by creating new business models and improve productivity, exploit analytics for innovation, and transform workforce. Among all the technologies, predictive maintenance is probably a relatively "easier win" since it is applicable to existing assets and management systems. The objective of intelligent maintenance systems is to reduce unexpected downtime and increase productivity. Cyber-physical systems can be designed by following the 5C connection, conversion, cyber, cognition, configuration architecture, [65] and it will transform the collected data into actionable information, and eventually interfere with the physical assets to optimize processes. However, without sensing and intelligent analytics, it can be only determined by experience when the band saw belt will actually break. The developed prognostics system will be able to recognize and monitor the degradation of band saw belts even if the condition is changing, advising users when is the best time to replace the belt. This will significantly improve user experience and operator safety and ultimately save on costs. This data can be used to automate farming techniques, take informed decisions to improve quality and quantity, minimize risk and waste, and reduce effort required to manage crops. For example, farmers can now monitor soil temperature and moisture from afar, and even apply IoT-acquired data to precision fertilization programs. Developed in part by researchers from Kindai University , the water pump mechanisms use artificial intelligence to count the number of fish on a conveyor belt , analyze the number of fish, and deduce the effectiveness of water flow from the data the fish provide. IoT can benefit the construction industry by cost saving, time reduction, better quality workday, paperless workflow and increase in productivity. It can help in taking faster decisions and save money with Real-Time Data Analytics. It can also be used for scheduling repair and maintenance activities in an efficient manner, by coordinating tasks between different service providers and users of these facilities. Usage of IoT devices for monitoring and operating infrastructure is likely to improve incident management and emergency response coordination, and quality of service , up-times and reduce costs of operation in all infrastructure related areas. For example, Songdo , South Korea, the first of its kind fully equipped and wired smart city , is gradually being built, with approximately 70 percent of the business district completed as of June [update]. Much of the city is planned to be wired and automated, with little or no human intervention. For this deployment, two approaches have been adopted. This city of , inhabitants has already seen 18, downloads of its city smartphone app. The app is connected to 10, sensors that enable services like parking search, environmental monitoring, digital city agenda, and more. City context information is used in this deployment so as to benefit merchants through a spark deals mechanism based on city behavior that aims at maximizing the impact of each notification. The network was designed and engineered by Fluidmesh Networks, a Chicago-based company developing wireless networks for critical applications. With the wireless network in place, NY Waterway is able to take control of its fleet and passengers in a way that was not previously possible. New applications can include security, energy and fleet management, digital signage, public Wi-Fi, paperless ticketing and others. IoT devices in this application typically span a large geographic area and can also be mobile. IoT creates opportunities for more direct integration of the physical world into computer-based systems, resulting in efficiency improvements, economic benefits, and reduced human exertions. Ambient intelligence and autonomous control do not necessarily require Internet structures, either. However, there is a shift in research by companies such as Intel to integrate the concepts of IoT and autonomous control, with initial outcomes towards this direction considering objects as the driving force for autonomous IoT. Modern IoT products and solutions in the marketplace use a variety of different technologies to support such context-aware automation, but more sophisticated forms of intelligence are requested to permit sensor units and intelligent cyber-physical systems to be deployed in real environments. The specific problem is: The information is partially outdated, unclear, and uncited. WikiProject Technology may be able to help recruit an expert. July IIoT system architecture, [] in its simplistic view, consists of three tiers: Tier 2 includes sensor data aggregation systems called Edge Gateways that provide functionality, such as pre-processing of the data, securing connectivity to cloud, using systems such as WebSockets, the event hub, and, even in some cases, edge analytics or fog computing. Tier 3 also includes storage of sensor data using various database systems, such as time series databases or asset stores using backend data storage systems such as Cassandra or Postgres. In addition to the data storage, we analyze the data using various analytics, predictive or threshold-based or regression-based, to get more insights on the IIoT equipment. Building on the Internet of things, the web of things is an architecture for the application layer of the Internet of things looking at the convergence of data from IoT devices into Web applications to create innovative use-cases. In order to program and control the flow of information in the Internet of things, a predicted architectural direction is being called BPM Everywhere which is a blending of traditional process management with process mining and special capabilities to automate the control of large numbers of coordinated devices. With billions of devices [] being added to the Internet space, IPv6 will play a major role in handling the network layer scalability. Fog computing is a viable alternative to prevent such large burst of data flow through Internet. At the overall stage full open loop it will likely be seen as a chaotic environment since systems always have finality. As a practical approach, not all elements in the Internet of things run in a global, public space. Subsystems are often implemented to mitigate the risks of privacy, control and reliability. For example, domestic robotics domotics running inside a smart home might only share data within and be available via a local network. Human beings in surveyed urban environments are each surrounded by to trackable objects. This number is about to grow up to million devices in and will for sure go on growing in the near future. Note that some things in the Internet of things will be sensors, and sensor location is usually important. However, the challenges that remain include the constraints of variable spatial scales, the need to handle massive amounts of data, and an indexing for fast search and neighbor operations. In the Internet of things, if things are able to take actions on their own initiative, this human-centric mediation role is eliminated. Thus, the time-space context that we as humans take for granted must be given a central role in this information ecosystem. Just as standards play a key role in the Internet and the Web, geospatial standards will play a key role in the Internet of things. Others are turning to the concept of predictive interaction of devices, "where collected data is used to predict and trigger actions on the specific devices" while making them work together. Crucial to the field is the network used to communicate between devices of an IoT installation, a role that several wireless or wired technologies may fulfill: The objects themselves do not converse, but they may now be referred to by other agents, such as powerful centralized servers acting for their human owners. Due to the limited address space of IPv4 which allows for 4. To a large extent, the future of the Internet of things will not be possible without the support of IPv6; and consequently, the global adoption of IPv6 in the coming years will be critical for the successful development of the IoT in the future. Light-Fidelity Li-Fi â€” Wireless communication technology similar to the Wi-Fi standard, but using visible light communication for increased bandwidth. QR codes and barcodes â€” Machine-readable optical tags that store information about the item to which they are attached. Radio-frequency identification RFID â€” Technology using electromagnetic fields to read data stored in tags embedded in other items. Transport Layer Security â€” Network security protocol. Medium-range wireless[edit] LTE-Advanced â€” High-speed communication specification for mobile networks. Provides enhancements to the LTE standard with extended coverage, higher throughput, and lower latency. Long-range wireless[edit] Low-power wide-area networking LPWAN â€” Wireless networks designed to allow long-range communication at a low data rate, reducing power and cost for transmission.

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

3: Tutorial: Train an image classification model with Azure Machine Learning service | Microsoft Docs

Retrieve Azure Resource Manager virtual machine properties by using PowerShell - Part 4 Retrieve Azure Resource Manager virtual machine properties by using PowerShell - Part 5 After you finish this series, move on to the next series to learn more about the Azure Resource Manager cmdlets.

Create a directory to deliver the necessary code from your computer to the remote resource. Run the following code to create the training script called train. This training adds a regularization rate to the training algorithm, so produces a slightly different model than the local version. The training script reads an argument to find the directory containing the data. When you submit the job later, you point to the datastore for this argument: Copy this script into the script folder so that it can be accessed along with the training script on the remote resource. Create your estimator by running the following code to define: The name of the estimator object, est The directory that contains your scripts. All the files in this directory are uploaded into the cluster nodes for execution. In this case you will use the Batch AI cluster you created The training script name, train. All files in the project directory are uploaded into the cluster nodes for execution. Monitor a remote run In total, the first run takes approximately 10 minutes. A Docker image is created matching the Python environment specified by the estimator. The image is uploaded to the workspace. Image creation and uploading takes about 5 minutes. This stage happens once for each Python environment since the container is cached for subsequent runs. During image creation, logs are streamed to the run history. You can monitor the image creation progress using these logs. If the remote cluster requires more nodes to execute the run than currently available, additional nodes are added automatically. Scaling typically takes about 5 minutes. While the job is running, stdout and the. You can check the progress of a running job in multiple ways. Jupyter widget Watch the progress of the run with a Jupyter widget. Like the run submission, the widget is asynchronous and provides live updates every seconds until the job completes. Get log results upon completion Model training and monitoring happen in the background. Wait until the model has completed training before running more code. Retrieve the accuracy of the model: This content appears in the run record in the experiment under your workspace. Hence, the model file is now also available in your workspace. You can see files associated with that run. In the Azure portal, select Resource groups on the far left. From the list, select the resource group you created. Select Delete resource group. Enter the resource group name, and then select Delete. If you see the error message "Cannot delete resource before nested resources are deleted," you must delete any nested resources first. For information on how to delete nested resources, see this troubleshooting section. You can also delete just the Azure Managed Compute cluster. However, since autoscale is turned on and the cluster minimum is 0, this particular resource will not incur additional compute charges when not in use. Set up your development environment Access and examine the data Train a simple logistic regression locally using the popular scikit-learn machine learning library Train multiple models on a remote cluster Review training details and register the best model You are ready to deploy this registered model using the instructions in the next part of the tutorial series:

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

4: JavaScript tutorial

Hi Hela, well, I'm no expert (I'm new to this forum), but I think that "at the beginning" implies a reference to a more specific point in time, where as "in the beginning" implies something occuring over a slightly longer period of time.

For many who work in social media, one of the hardest tasks is creating content that adds value days a year. If you work in sports though, thankfully, that is normally NOT the problem: While I was writing the article though, I naturally kept coming back to how user-generated content could be leveraged in the sports industry. Just like a lot of brands outside of sports, I normally see it used for contests. It gives you more content, while being cost effective. Leveraging user-generated content is a great way to get more content while not dipping into your budget. It shows a different perspective. UGC could have a lot of leverage for teams and leagues on game days, providing unique perspective only fans can provide. It connects fans even more to the community. Additionally, people are more likely to share content they are a part of. Quite simply, it makes your community stronger and is an easy way to thank your fans. So are you convinced that user-generated content has value, but not sure exactly what that looks like? How do you source user-generated content beyond contests? There are two key ways to gather user-generated content: Soliciting user-generated content on owned channels is common practice in the fashion industry. Many of my favorite brands allow consumers to upload photos of them in the item and leave a review on the product. Free People is a great example of it here. Teams and leagues can replicate this model from the fashion world and solicit user-generated photos on their owned channels year-round. If you want to solicit user-generated content on owned channels year-round, but are afraid it might get stale, think about setting themes of what you are looking for around seasons, sports, themes, etc. Let fans know what type of content you are looking for, give them hashtags to rally around, ask them to leave photos in the comments section of Facebook, etc. Actively seek it out. One of the many benefits of social media is the ability to listen, see and engage with your consumers in real-time. When you see something that strikes your fancy, just ask if you can use it. Now make it your own. One of the biggest mistakes made with user-generated content is simply retweeting, regramming, etc. If you have permission to the use the content, then make it your own instead. This is the key to using UGC on a more regular basis and making sure it resonates with your community. Just be sure you always thank and call out the fans that helped to make the post come to life. But, what does this look like? Sometimes it helps to have examples to see the possibilities. Being a one-person social media team, there is no way that I can be everywhere at once to cover all the schools. This is where user-generated content comes in to help! How can UGC fit into the story we are trying to tell? The graphics below are not from the SEC. I created them for demonstration purposes: What if the SEC wants to promote selfies yeah, sorry: If you want to think really big, you can use fan-generated video to create a compelling compilation like Baylor Athletics did thanks for the find, Katie Cavendar or UCF the videos from Instagram were saved from InstaSave: As you can see, the options are endless. Both teams and leagues can leverage user-generated content to tell the story of traditions, fan perspectives, in-stadium experience, view from the student section, spots around campus, selfies with mascots and so much more. Fans are already out there creating great content for teams and leagues: Do you have any great examples of teams and leagues leveraging user-generated content? If so, share them below! Like what you read?

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

5: Machine | www.enganchecubano.com

Note: Only the SSA for the main function will be shown, as that is the interesting part. To show the generated SSA, we will need to set the GOSSAFUNC environment variable to the function we would like to view the SSA of, in this case main.

This will work cross browser. However, just be aware that you may encounter some of these, especially in text inputs Mac OS 9 is still in use among some Mac users. If you are writing a string that contains double quotes, the easiest way to deal with it is to surround it with single quotes. You should also use double quotes if your string contains single quotes: What if you have both single and double quotes in your string? For example, both of the following are valid: Netscape 4 inventors of JavaScript , Escape, Opera 7. ICEbrowser sometimes ignores the whole string. I mention this only for completeness. Do not use it. It is easier and better to use the concatenation operator as shown above. Regular expressions Opera has a major bug in its regular expression handling and will fail on even the simplest regular expressions. Certain methods, like the stringObject. To do this, they use regular expressions. These are well documented elsewhere so I will not describe the syntax here. Some rare browsers do not support regular expressions. In theory, you should be able to detect regular expression support using this: However, as Opera 6- and iCab 2- support the RegExp object, but fail to use it correctly, your scripts will still fail, though hopefully without errors. Thankfully, current releases of Opera and iCab do support regular expressions correctly. If it had been anything other than 2 it would now be 0. For example, some browsers provide document. In order to solve this problem, we could write this: This is known as a short-circuit. The operator has a similar feature, but it will only evaluate the second test if the first one fails. This may seem convenient, as it allows you to make your code a tiny bit shorter, but I recommend avoiding this syntax. It makes your code harder to read, especially if you start nesting your control structures. It also makes it easy to forget to put them in when you needed them, and also makes debugging code much harder, since you will need to go back through your code to add them so that you can add extra debugging tests. It is best to always use the curly braces, even if they are optional. As always, there is an exception. Typically, it is used to cycle through the contents of an array, or to create a specific number of new objects, but it can do many more useful things if needed. As with all variables, you must declare them if you have not done so already. You can define multiple variables if needed, using: Typically, this is used to increment or decrement a stepping variable, and it is possible to perform actions on more than one variable by separating them with a comma: Every time you create properties or methods on an object, these will be added to the list of properties that will be exposed. Most internal properties the ones that JavaScript creates will also be exposed, but JavaScript engines are allowed to hide internal properties and methods if they want to. You should not rely on any specific behaviour here, but note that some browsers will give the internal properties and methods of intrinsic objects, and some will not. Again, you should declare the variable names that you use, if you have not done so already. Each time it loops, it assigns the next property name as a string value to myVariable. You can then use array notation to access the value of that property. The following example writes all the exposed properties of the document object: It is very easy to make mistakes here, so be careful not to mistake these property types for each other. It will continue to run as long as the condition is satisfied: Firstly, it would use the value of myVariable to index the array cell, then it would increment myVariable. Firstly, it would increment the value of myVariable, then it would use the new value to index the array cell. If I had done this, myArray[2] to myArray[6] would now be 1. These features also work outside loops, but this is where you will most commonly see them, so I have included them here. The condition is evaluated at the end of the loop, meaning that even if the condition is never satisfied, it will still run through the loop at least once. The solution is to use break; as follows The use of the break statement is described below. This makes it superior to the original way of handling script errors without error messages where scripts are completely aborted: Thankfully these browsers are hardly used any more. It would also be useful for checking for stupid bugs, like where checking for something like navigator. However, the error is not correctly thrown for these errors. Unfortunately, if you use this structure in any script run by a browser that does not support it, the browser will abort the entire script with errors, even if it does not use the part containing the structure. Thankfully, these old browsers can be safely ignored. It should never be used to detect if a browser supports a method or property like document. So when should it be used? It can be used for W3C DOM scripting, where you may want to avoid DOM mutation errors for example , which are valid errors, but serve to warn you not to do something, and do not always need to abort the whole script. However, they will still run the script it is not possible to protect them by using the language attribute on the script tag, as you need to use JavaScript 1. This means that the older browsers will still produce errors, unless you define the old error handling method in an earlier script. It can be used to check if accessing a frameset frame will cause a browser security error for example, if the page in the frame belongs to another site. It could also enable you to avoid problems where different browsers support the same methods but expect a different syntax, for example, the selectBox. This is a very important rule to learn, as it forms the basis of object and capability detection, and is fundamental to making cross browser scripts. This will be true if: Netscape 4, Internet Explorer 5. It can also be used to check for the existence of named properties of an object. In most cases, it is best to use a conditional without a condition, as shown above. An example would be where you want to check for the existence of a property whose value may be 0, or an empty string, or null. If you know what the type of the property will be, it is possible to achieve this using identity operators, or the typeof operator, as shown here: This allows you to test for its existence, no matter what value it currently holds, and no matter what type of value it currently has even if it has been assigned a value of undefined. This limits the usefulness a little, as it can only search for the name, and cannot be used to see if one of the properties holds a specific value or value type. In most other browsers, the two alternatives perform about the same. Assignments inside a conditional JavaScript allows you to perform an assignment at the same time as testing if the assignment worked. The syntax used is: Control Structure Labels are very rarely used in JavaScript. The break statement Writing break inside a switch, for, for-in, while or do - while control structure will cause the program to jump to the end of the statement. If you just use, for example: But if you use this: If you just use this, for example: If you use this instead: Writing with script Be careful when writing with script. If script is not available, that content will not be created. You should limit your use of this to parts of the page that are not needed to access page content. Writing while the page is still loading If your code is being executed while the page is still loading in other words, if it is run as part of the initial page layout put the following: Of course, you can include HTML tags in there too or some pre-programming. Note that if you write content using an event handler such as the onload handler for an image , it will be treated as if the page has completed loading, even if it has not. Writing after the page has loaded After the page has completed loading, the rules change. Instead of adding content to the page, it will replace the page. To do this, you should firstly open the document stream most browsers will automatically do this for you if you just start writing. Then you should write what you want, and finally, you should close the document stream. Again, most browsers will automatically close the stream for you. Some other browsers may fail to render part or all of the content. Just to be safe, make sure you always close the stream. This is the equivalent of moving the user to a completely new page. You may notice that I close my HTML tags inside the script with a backslash before the forward slash in the closing tag. This is a requirement of the specification and can cause the HTML validator not to validate your page if you forget it , although all browsers will understand if you omit the backslash. However, since you can write HTML with script, you can write style or even script tags with it, making one script import another. The same applies to opening or closing comments although I fail to see why you would want to write comments using a script. When the script runs, the plus sign tells it to append the strings, creating a valid HTML comment. Problems with old browsers Although not in use any more, you may want to be nice to older browsers like Netscape 4.

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

6: How debuggers work: Part 3 - Debugging information - Eli Bendersky's website

Bendy is a cartoon character created by the animation company Joey Drew Studios as their mascot, and the titular character in the Bendy and the Ink Machine series. The studio's flagship character, Bendy starred in a series of cartoons thirty years prior to the events of the first game, often alongside the studio's other characters such as Boris the Wolf and Alice Angel.

The high standard of living in the developed countries owes much to mechanical engineering. The mechanical engineer invents machines to produce goods and develops machine tools of increasing accuracy and complexity to build the machines. Machine components in an automobile As part of an introduction to machine components, some examples supplied by an automobile are of value. In an automobile, the basic problem is harnessing the explosive effect of gasoline to provide power to rotate the rear wheels. The explosion of the gasoline in the cylinders pushes the pistons down, and the transmission and modification of this translatory linear motion to rotary motion of the crankshaft is effected by the connecting rods that join each piston to the cranks that are part of the crankshaft. The piston, cylinder, crank , and connecting rod combination is known as a slider-crank mechanism; it is a commonly used method of converting translation to rotation as in an engine or rotation to translation as in a pump. To admit the gasolineâ€”air mixture to the cylinders and exhaust the burned gases, valves are used; these are opened and closed by the wedging action of cams projections on a rotating camshaft that is driven from the crankshaft by gears or a chain. In a four-stroke-cycle engine with eight cylinders, the crankshaft receives an impulse at some point along its length every quarter revolution. To smooth out the effect of these intermittent impulses on the speed of the crankshaft, a flywheel is used. This is a heavy wheel, attached to the crankshaft, that by its inertia opposes and moderates any speed fluctuations. Since the torque turning force that it delivers depends on its speed, an internal-combustion engine cannot be started under load. To enable an automobile engine to be started in an unloaded state and then connected to the wheels without stalling, a clutch and a transmission are necessary. The former makes and breaks the connection between the crankshaft and the transmission, while the latter changes, in finite steps, the ratio between the input and output speeds and torques of the transmission. In low gear , the output speed is low and the output torque higher than the engine torque, so that the car can be started moving; in high gear, the car is moving at a substantial speed and the torques and speeds are equal. The axles to which the wheels are attached are contained in the rear axle housing, which is clamped to the rear springs, and are driven from the transmission by the drive shaft. As the car moves and the springs flex in response to bumps in the road, the housing moves relative to the transmission; to permit this movement without interfering with the transmission of torque, a universal joint is attached to each end of the drive shaft. The drive shaft is perpendicular to the rear axles. The right-angled connection is usually made with bevel gears having a ratio such that the axles rotate at one-third to one-fourth the speed of the drive shaft. The rear axle housing also holds the differential gears that permit both rear wheels to be driven from the same source and to rotate at different speeds when turning a corner. Like all moving mechanical devices, automobiles cannot escape from the effects of friction. In the engine, transmission, rear axle housing, and all bearings, friction is undesirable, since it increases the power required from the engine; lubrication reduces but does not eliminate this friction. On the other hand, friction between the tires and the road and in the brake shoes makes traction and braking possible. The belts that drive the fan , generator, and other accessories are friction-dependent devices. Friction is also useful in the operation of the clutch. Some of the devices cited above are found in machines of all categories, assembled in a multitude of ways to perform all kinds of physical tasks. The function of most of these basic mechanical devices is to transmit and modify force and motion. Other devices, such as springs, flywheels, shafts, and fasteners , perform supplementary functions. A machine may be further defined as a device consisting of two or more resistant, relatively constrained parts that may serve to transmit and modify force and motion in order to do work. The requirement that the parts of a machine be resistant implies that they be capable of carrying imposed loads without failure or loss of function. Although most machine parts are solid metallic bodies of suitable proportions, nonmetallic materials, springs, fluid pressure organs, and tension organs such as belts are also employed. Constrained motion The most distinctive characteristic of a machine is that the parts are interconnected and guided in such a way that their motions relative to one another are constrained. Relative to the block, for example, the piston of a reciprocating engine is constrained by the cylinder to move on a straight path; points on the crankshaft are constrained by the main bearings to move on circular paths; no other forms of relative motion are possible. On some machines the parts are only partially constrained. If the parts are interconnected by springs or friction members, the paths of the parts relative to one another may be fixed, but the motions of the parts may be affected by the stiffness of the springs, friction, and the masses of the parts. If all the parts of a machine are comparatively rigid members whose deflections under load are negligible, then the constrainment may be considered complete and the relative motions of the parts can be studied without considering the forces that produce them. For a specified rotational speed of the crankshaft of a reciprocating engine, for example, the corresponding speeds of points on the connecting rod and the piston can be calculated. The determination of the displacements, velocities, and accelerations of the parts of a machine for a prescribed input motion is the subject matter of kinematics of machines. Such calculations can be made without considering the forces involved, because the motions are constrained. Mechanism of a machine According to the definition, both forces and motions are transmitted and modified in a machine. The way in which the parts of a machine are interconnected and guided to produce a required output motion from a given input motion is known as the mechanism of the machine. The piston, connecting rod, and crankshaft in a reciprocating engine constitute a mechanism for changing the rectilinear motion of the piston into the rotary motion of the crankshaft. Although both forces and motions are involved in the operation of machines, the primary function of a machine may be either the amplification of force or the modification of motion. A lever is essentially a force increaser, while a gearbox is most often used as a speed reducer. The motions and forces in a machine are inseparable, however, and are always in an inverse ratio. The output force on a lever is greater than the input force, but the output motion is less than the input motion. Similarly, the output speed of a gear reducer is less than the input speed, but the output torque is greater than the input torque. In the first case a gain in force is accompanied by a loss in motion, while in the second case a loss in motion is accompanied by a gain in torque. Although the primary function of some machines can be identified, it would be difficult to classify all machines as either force or motion modifiers; some machines belong in both categories. All machines, however, must perform a motion-modifying function, since if the parts of a mechanical device do not move, it is a structure, not a machine. While all machines have a mechanism, and consequently perform a motion-modifying function, some machines do not have a planned force-modifying purpose; the forces that exist are caused by friction and the inertia of the moving masses and do not appear as a useful output effort. This group would include measuring instruments and clocks. If a man carries a weight along a horizontal path, he does no work according to this definition, since the force and the motion are at right angles to one another; that is, the force is vertical and the motion horizontal. If he carries the weight up a flight of stairs or a ladder, he does work, since he is moving in the same direction in which he is applying a force. When a force causes a body to rotate about a fixed axis, or pivot, the work done is obtained by multiplying the torque T by the angle of rotation. Calculating efficiency These concepts of work are fundamental in defining the mechanical work function of machines in terms of forces and motions, and they bring out the inseparability of forces and motions in machines. Because of friction, the work output from a machine is always less than the work input, and the efficiency , which is the ratio of the two, is always less than percent. The ratio of the output to input forces is the mechanical advantage MA , and it defines the force-modifying function, while the ratio of the input to output motions is the velocity ratio VR , and it defines the motion-modifying function. When the efficiency is high, these ratios are approximately equal; if the output force is 10 times the input force, the input motion must be 10 times the output motion; i. Friction affects the mechanical advantage but not the velocity ratio except in some mechanisms using belts and idler pulleys. To calculate the efficiency from the ratio of output to input work, it would be necessary to know the work done by the output and input forces over a specified distance. Since this would entail the determination of average forces over the interval, it would be inconvenient. The efficiency of a machine is more easily determined from instantaneous values of load and the rate at which the load is moving. For this purpose, power formulas are most useful. Power is the rate at which work is done. If a man carries a pound 4. Learn More in these related Britannica articles:

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

7: Web - Wikipedia

Note: Contents data are machine generated based on pre-publication provided by the publisher. Contents may have variations from the printed book or be incomplete or contain other coding. Contents may have variations from the printed book or be incomplete or contain other coding.

This is the third part in a series of articles on how debuggers work. Make sure you read the first and the second parts before this one. Debugging information Modern compilers do a pretty good job converting your high-level code, with its nicely indented and nested control structures and arbitrarily typed variables into a big pile of bits called machine code, the sole purpose of which is to run as fast as possible on the target CPU. Most lines of C get converted into several machine code instructions. Variables are shoved all over the place - into the stack, into registers, or completely optimized away. So how does a debugger know where to stop when you ask it to break at the entry to some function? How does it manage to find what to show you when you ask it for the value of a variable? The answer is - debugging information. Debugging information is generated by the compiler together with the machine code. It is a representation of the relationship between the executable program and the original source code. This information is encoded into a pre-defined format and stored alongside the machine code. Many such formats were invented over the years for different platforms and executable files. DWARF is a complex format, building on many years of experience with previous formats for various architectures and operating systems. It has to be complex, since it solves a very tricky problem - presenting debugging information from any high-level language to debuggers, providing support for arbitrary platforms and ABIs. In this article I will take a more hands-on approach, showing just enough of DWARF to explain how debugging information works in practical terms. ELF defines arbitrary sections that may exist in each object file. A section header table defines which sections exist and their names. Different tools treat various sections in special ways - for example the linker is looking for some sections, the debugger for others. The debugger uses this information to read the section from the executable. Finding functions One of the most basic things we want to do when debugging is placing breakpoints at some function, expecting the debugger to break right at its entrance. To be able to perform this feat, the debugger must have some mapping between a function name in the high-level code and the address in the machine code where the instructions for this function begin. Before we go further, a bit of background. Each DIE has a tag - its type, and a set of attributes. This is the program-counter EIP in x86 value for the beginning of the function. How does it know where to find it? Turns out this is much trickier than finding functions. Variables can be located in global storage, on the stack, and even in registers. Additionally, variables with the same name can have different values in different lexical scopes. For each address where the debugger may be, it specifies the current frame base from which offsets to variables are to be computed as an offset from a register. For x86, bpreg4 refers to esp and bpreg5 refers to ebp. We get ebp - Looking up line numbers When we talked about finding functions in the debugging information, I was cheating a little. This is why DWARF encodes a full mapping between lines in the C source code and machine code addresses in the executable. This information is contained in the. File name Line number Starting address tracedprog2. This line information easily allows bi-directional mapping between lines and addresses: When asked to place a breakpoint at a certain line, the debugger will use it to find which address it should put its trap on remember our friend int 3 from the previous article? When an instruction causes a segmentation fault, the debugger will use it to find the source code line on which it happened. Now, remember how everyone keeps saying that you should never, ever parse HTML manually but rather use a library? Since libdwarf is itself quite complex it requires a lot of code to operate. The demonstrated program takes an executable and prints the names of functions in it, along with their entry points. Conclusion and next steps Debugging information is a simple concept in principle. With this information in hand, the debugger bridges between the world of the user, who thinks in terms of lines of code and data structures, and the world of the executable, which is just a bunch of machine code instructions and data in registers and memory. This article, with its two predecessors, concludes an introductory series that explains the inner workings of a debugger. Using the information presented here and some programming effort, it should be possible to create a basic but functional debugger for Linux. Readers can also suggest ideas for future articles in this series or related material. Feel free to use the comments or send me an email.

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

8: Bendy | Bendy and the Ink Machine Wiki | FANDOM powered by Wikia

Note: Citations are based on reference standards. However, formatting rules can vary widely between applications and fields of interest or study. The specific requirements or preferences of your reviewing publisher, classroom teacher, institution or organization should be applied.

You will find it here: Open your Azure DevOps organization eg. Click Create and you will see the token one time and one time only. You will need to copy the token. You can use the same token for multiple agents. For the automated build agent setup, open http: After minutes, you should see your new agent popping up https: Access your secrets in the build system In part 1, we created an Azure KeyVault, with a number of secrets password, license file etc. We need these secrets in the build and fortunately, Azure DevOps has a nice way of getting to those. Name the variable group and select Link secrets from Azure key vault as variables. Specify the subscription you will need to authenticate , select the key vault and add the variables These variables will now be available for the build tasks if you link the variable group to the pipeline. You will likely see that you have no build pipelines: Click New pipeline and click Use the visual designer. Select the Default Agent pool and select the CI. You can click the link of the build to see the log The first time you run a build on the agent, it will take significantly longer because it needs to pull the Business Central Docker image. After some time, you should see your build successfully completed and one failing test. A few words about Tests I should mention here, that the way tests are executed in the repository while writing this blog post is still work in progress. We are working on making a better approach to this and I will include this in a later blog post. In the next Blog Post we will setup checkin policies with code reviews and pull requests - and we will create a few more pipelines for testing our app against insider builds.

9: Citation Machine: Format & Generate Citations â€“ APA, MLA, & Chicago

Debugging information is generated by the compiler together with the machine code. It is a representation of the relationship between the executable program and the original source code. This information is encoded into a pre-defined format and stored alongside the machine code.

The word appeared in Latin as machina, from which it came into English. The principles of machines were known emprically from ancient times, and the theory was elaborated and put on a rational basis by Archimedes 3rd century BC. Simple machines have always been an elementary part of Physics instruction, and are of continuing importance in daily life. It is good to start with clear definitions, so that we know what we are talking about. Let us say that a machine is a collection of resistant bodies arranged to change the magnitude, direction or point of application of moving forces. Motion is an essential part of a machine; without it, at least in principle, we have no machine, but a structure. The restriction to resistant bodies sets hydraulic and other fluid machines aside; these deserve special treatment. Some authors classify the hydraulic press as a machine. In a sense it is, of course, and depends on statics, but we will leave it aside. An ideal machine is one in which the parts are considered to be weightless, frictionless and rigid. Real machines are not ideal, but ideal machines aid thought and analysis, and in many cases are adequate approximations, so they are quite useful. A simple machine is a machine from which no part can be removed without destroying it as a machine. A mechanism is a machine considered solely from the point of view of its motions, kinematically, without consideration of loads. Some authors say a mechanism is a machine with "that does no useful work," but this is not a helpful distinction. A steam engine valve gear is essentially a mechanism to obtain a particular motion, but it also does useful work on the valve. A structure transmits force without motion. It may be useful to define an engine as a machine in which the input is not in the form of mechanical energy, but which is converted into forces and torques by the machine. For example, the input could be electrical, or provided by a heat engine. It could also include machines worked by men or animals considered as part of the machine and not as users of it. A prime mover is an engine whose power is derived from some nonmechanical source, such as a heat engine. A prime mover is capable of motion, or being moved, without connection to any other system. However, windmills, water wheels and turbines are considered to be prime movers, as clearly are men and animals. Fundamentally, engine and machine are actually two words for the same thing, derived from Latin and Greek, respectively. There are many ways of transmitting forces in machines, such as fluids pressing on a piston in a cylinder, or exerting their weight in buckets, flexible agents such as ropes, belts, cables and chains, springs, and weights themselves. These means are not part of the machine itself. Weight is the force of gravity on massive bodies, and form a very common load on a machine. A link is a member that transmits an axial force of compression or tension, and is connected by pins or sliders at its ends. A link is not a machine by itself it does not transform its input , but is a typical part of a mechanism, and may transmit forces between simple machines. A slotted link with a sliding block may permit a variable amount of motion to be transmitted. Every machine has an input and an output, and the output is a modification of the input, not a simple replication of it. A machine is a processor or transformer in some sense. The motion of the output is fully constrained by the motion of the input, by their kinematic connection. The force at the input is called the effort, and the force at the output, a load. The mechanical advantage, which we shall call simply the advantage, is the ratio of the load to the effort. The velocity ratio is the ratio of the movement of the load to the movement of the effort, in linear displacement or rotation. In an ideal machine the product of the advantage and the velocity ratio is unity, as we shall see. There is a trade-off between force and speed. In a real machine the product is less than unity. As a consequence, an ideal machine in equilibrium when the effort and the load balance can be moved by the least impetus, as well in one direction as in the other, so the machine is reversible. A real machine, however, requires a certain effort to move it in either direction; it is irreversible, and there is an unavoidable loss of energy whenever it moves. It is reasonable to exclude from our definition those devices that depend essentially on inertial forces. The pendulum is one such device, as is the whole family of fluid turbines, and perhaps sails and airfoils as well. Simple machines can, however, form a part of such devices. These devices all deserve special consideration, and involve matters not essential to the machines that will be discussed here. Therefore, our machines depend only on the principles of statics and kinematics, not dynamics. Dynamics may have to be considered in connection with the design of machine elements, however. The inputs and outputs of a machine may be either forces or torques, and a machine may convert one into the other. A torque or moment tends to cause rotation, while a force causes linear motion. The work done is either torque times angle of rotation, or force times distance. The dimensions of torque are force times distance, and this should be carefully distinguished from work, which has the same dimensions. Sometimes, torque is stated in, for example, pound-foot while work is in foot-pound to make this clear. A fundamental property of machines is that the input and output work are the same, except for frictional losses that make the output work smaller. This principle of the conservation of energy is a very important generalization, and will be considered in more detail later. To understand the magnitudes of the forces in a machine, the methods of statics are used. If you already know statics, then the application to machines will be easy. If you do not, machines are an excellent and graphic way to learn about statics, and will help you to understand it. Briefly, we note that forces add according to the parallelogram rule, and can be resolved trigonometrically into components in many ways, the most useful being the rectangular components. The moment of a force about an axis is the product of the force and the shortest distance between its line of action and the axis. A body is in equilibrium if the sum of the forces acting upon it is zero, and the moment of these forces about any axis is zero. This gives up to six equations that may be used to find the magnitude and direction of unknown forces. In applying these principles, it is best to draw the body in question isolated from all others, and show the forces acting on it, and only those forces. Since ancient times, simple machines have been classified as lever, wedge, wheel and axle, pulley and screw. Sometimes the wedge and screw are considered special cases of the inclined plane, so there are either four or six simple machines. This is no more than an arbitrary and incomplete taxonomy. Since classifications should be useful, you should try to make your own classification that reminds you of the principal similarities and differences. I prefer to divide simple machines into three families, those of the lever, the inclined plane and the pulley, and will treat machines in that order in this paper. Each family has various tribes, and some tribes are descendants of two families. There is also a miscellaneous family in which mechanisms are put that fit nowhere else. In complex machines, the families are mixed and connected in glorious variety. There are ingeneous devices that, while not machines in themselves, are very important parts of machines. These include bearings, couplings, clutches, cams, springs and gears, which are conveniently studied in connection with machines. Real Machines Friction is usually the most important reason real machines are different from ideal ones, and in some machines friction plays an essential role. For example, belt drives would not work without friction. Friction cannot be described accurately in a few words, since it is a very complex and variable phenomenon. The area over which N is spread is of no significance. Once motion starts, the minimum force required to maintain motion is less than F, and may depend on the speed, usually decreasing with increasing speed. This is sliding friction. Rolling friction is much less, practically vanishing if the surfaces in contact are hard and smooth, and there is no deflection. Journal bearingsare good examples of minimizing sliding friction. The metals in contact must be different, or the bearing will seize, especially at high bearing pressures and with poor lubrication. One metal should be hard, the other soft. Many bearings use a soft metal babbitt in recesses in a harder metal to confine it, working against polished steel. Steel and brass are a common pair for journal bearings. Lubrication can be supplied by greases that form an adherent thin film between the surfaces, by graphite flakes, or other means. An excellent bearing uses oil forced between the mating surfaces, dragged in by the relative movement. Now the contact is metal-oil, and the coefficient of friction drops sharply. Such a bearing is called hydrodynamic. Ball and roller bearings, called anti-friction bearings, take advantage of rolling contact between the inner and outer races, and require only minimal lubrication. Their friction is comparable to that of a hydrodynamic bearing when the latter is moving, but they require far less maintenance, besides having low friction when starting from rest. Leonardo da Vinci sketched a supposed anti-friction bearing, shown in the Figure, that seems to have been the basis for several later attempts to avoid journal bearings. It does not work, of course, merely subdividing the friction. Rolling contact is essential. A second limiting factor in machines is the elasticity of their parts. If a machine is scaled up proportionately, the loads F increase as L3, the areas only as L2, so the deflection increases as the square of the size. Bending due to transverse forces leads to much larger deflections than simple tension or compression, and these deflections increase more rapidly with size. The effect is even greater in bending. Scaling a machine up in size must be done with care. Materials also crush under bearing pressure. Wood is particularly subject to crushing perpendicularly to the grain.

MACHINE GENERATED CONTENTS NOTE: PART I: IN THE BEGINNING pdf

Knight of the Basilisk The search for a common language The Invention of Infinity 1. The Action Principles in Mechanics Sakurai modern quantum mechanics 2nd edition solutions Responsibility in Washington The management of employment relations Needs assessment of motor proficiency and health-related fitness for children conducted in cooperation wi Players guide to the high clans 101 ways to have true love in your life Jewish Biblical commentators. Recent Progress in Intersection Theory I knew Id sing Heather McHugh St. James guide to Black artists Cryptography and network security solution manual 5th edition Can feminism be liberated from governmentalism? John Tomasi Society and education in England since 1800 The unbeaten Lions Fluent 6.3.26 user guide Pinkitys pranks and other nature fairy tales. The Subterranean Dungeon The philosophy of time travel book Privatisation, globalisation, and labour Tropical garden city Fundamentals of anatomy and physiology martini Wild foods of Great Britain The Speakers complete program Canadas prime ministers, Macdonald to Trudeau The contribution of critical scholarship to ministerial efficiency, by G. B. Foster. The Ethics of Cyberspace Gtb 2014 annual report Fire hd 8 user manual Biomechanics of the Hip Level of processing theory Sonamonider phycics Shikkha Larte Larte Phycics Shikha George Washington Carver (The Virginia Experience) The social course of St. Vituss dance Bernard Shaw and Alfred Douglas, a correspondence Love unconditionally The Cassandra problem