LANGUAGES OF CLASS pdf

1: World Class Languages

In Languages of Class Gareth Stedman Jones draws a distinction between two conceptions of class: the everyday and commonplace perception of its pervasiveness in England, and the Marxist idea of its revolutionary significance.

In this course, we will examine the language of deception and humor from a variety of perspectives: We will focus on the linguistic knowledge and skills that underlie the use of humor and deception and on what sorts of things they are used to communicate. A survey of linguistic phenomena typical of the Algonquian family of languages, including animacy-based gender, obviation, inverse verbs, deixis, noun incorporation, complex predicates, discontinuous constituents, separable preverbs, discourse conditions on word order, and templatic inflectional morphology. This course satisfies the non-Indo European requirement for undergraduate Linguistics major. LING or consent of instructor. This seminar focuses on current research in contact linguistics in a global perspective, including but not limited to the impact of languages of wider communication e. English, Russian in contact with other languages. Topics to be covered include the following: Where our ideas have come from, and where we think they have come fromâ€"these concerns have a powerful influence on the work that we do, and nowhere is this more true than in the academic fields that we call the mind sciences, which include linguistics, psychology, philosophy, and logic. This course will focus on several important moments in the developments in these fields, as viewed from the vantage point of a linguist in the 21st century. The course is based on a book, Battle in the Mind Fields, which I have written with Bernard Laks, and which will appear this summer from the University of Chicago Press. This book covers the era from the beginning of historical linguistics in Europe in the early 19th century, all the way up to the political turbulence in the s and s that led to World War II and the shift of the center of intellectual mass in the West from Europe to the United States. In this class we will cover computational techniques for collecting linguistic data. We will also cover various methods for using algorithms to analyze that data and some basic computational theory to understand the complexity and efficiency of our algorithms. We will use the programming language Python and focus on real-world applications to gain experience in gathering, manipulating, and analyzing data from sources such as field-work, corpora, or experiments. No previous knowledge of programming is required. This course studies the structure of Hungarian, a non-Indo-European language spoken in the heart of Europe. The course objectives are for students a to learn about the major grammatical properties of Hungarian and b to see the ways that Hungarian has come to shape the development of contemporary linguistic theory. The final two weeks of the quarter will be reserved for discussion of topics of interest to the students, such as quantification or historical change. This course satisfies the non-Indo-European language requirement for Linguistics majors and Linguistics graduate students. Undergraduates should have taken Introduction to Linguistics and, if possible, one of the following: Introduction to Phonology, Introduction to Syntax, or Morphology. Spoken in ten countries of Eastern and Central Africa, Swahili has more speakers than any other language in the Bantu family, a group of more than languages most prevalent in sub-equatorial Africa. Based on Swahili Grammar and Workbook, this course helps the students master key areas of the Swahili language in a fast yet enjoyable pace. Topics include sound and intonation patterns, noun class agreements, verb moods, and sentence structures. Additionally, this course provides important listening and expressive reading skills. For advanced students, historical interpretations are offered for exceptional patterns observed in Swahili, in relation with other Bantu languages. This is a general introduction course with no specific prerequisites. One third of world languages are spoken in Africa, making it an interesting site for studying linguistic diversity and language evolution. This course presents the classification of different African language families and explains their historical development and interactions. It also presents the most characteristic features of African languages, focusing on those that are common in Africa but uncommon among other world languages. Additionally, the course addresses the issue of language dynamics in relation to socioeconomic development in Africa. Using living audio and written material, students will familiarize themselves with at least one language of their choice. This is a course in the Computer Science department, intended for upper-level undergraduates, or graduate students, who have good programming skills. There will be weekly programming assignments in Python. We will look at several current topics in natural language processing, and discuss both the theoretical basis for the work and engaging in hands-on practical experiments with linguistic corpora. In line with most current work, our emphasis will be on systems that draw conclusions from training data rather than relying on the encoding of generalizations obtained by humans studying the data. As a consequence of that, in part, we will make an effort not to focus on English, but to look at a range of human languages in our treatments.

2: Language | Define Language at www.enganchecubano.com

Gareth Stedman Jones's Languages of Class is a collection of essays, which as the subtitle makes clear, studies the English working class at various moments from to The unifying theme of these essays is a concern with the big "C" issues of history: Class, Culture, Control, Consciousness and Custom.

When we started working with Logo in TG, all we had to do was enter commands and things happened. But with Java, we had to create a class with a main procedure called a method in Java. If Java programs are built from things called classes, what is a class? A class is a specification think of it as a blueprint or pattern and a set of instructions of how to provide some service. Like a blueprint or a pattern, a Java class has exact specifications. First, back when I introduced Adding Commands in Logo and again when we learned about Defining Operators, when we specified a contract for a Grid Toolkit. Engineers and construction and factory workers use blueprints to build cars, buildings, bridges, virtually anything. Tailors, seamsters, printers use patterns to make the clothes we wear, books we read. Chefs follow recipes to put together meals. Start by thinking of a class as an object. Imagine a class as a container, like that glass or plastic bottle you drink your favorite beverage from. Even something this simple has specifications - it has a size. It can only hold so much. A beverage company puts something in it. You take it out. More commonly, classes can store things and do things. Think about a modern digital camera as an object. You take pictures with it - it captures images and stores them. They also let you do things with the stored images, like cropping and stitching. Both of these objects, a bottle and a camera, can be represented as classes. You construct a class by writing what looks a lot like an outline. There are rules that you must follow to write this outline in the Java language, many of them and they are unambiguous. Logo had its own syntax. Its syntax is pretty simple, which is one reason I chose to expose you to it before Java. Java has a much more complex syntax. The rest of this lesson will cover the rules for writing a syntactically correct Java class. Why is this important? If you follow these rules, the Java compiler will not spit out a bunch of errors. Life will be good. A class Consists of Here is a breakdown of the source code representation of a Java class. A class can be broken down into two things: The first piece is a class-header which consists of the keyword "class" and the name you give to the class. Names in programming languages are also known as identifiers. The second piece is the body. Here is a template for the source code of a class: You can use this example every time you write the source code for a class. Specifically, the keyword "class" should start in the first column of a line followed by the rest of the header information - an identifier in the simplest case. Some programmers like to place it at the end of the header line; I like to place it on its own line, all by itself. A common practice is to place a close-squiggly-bracket in column 1 by itself. Members consist of variables called fields and things it does, called methods also known as procedures. Many programmers indent four columns at a time to make the source code easier to read. The number of columns you indent is not important, but you need to be consistent with whatever amount you choose. See indentation conventions in the jargon appendix for a longer explanation of why. There is another convention that you need to be aware of: Compare the class template above with the Hello World program in the last lesson. This method main is needed in every application. Java Keywords Java keywords are words reserved as core building blocks of the Java language. You may not use them for names of things you create, like class, field and method identifiers. The Java compiler recognizes these words and treats them special. The special meaning of these keywords will be covered as each is introduced in a lesson.

3: Class (computer programming) - Wikipedia

This collection of essays by Gareth Stedman Jones proposes a different way of seeing both historians' analytical conceptions of 'class', and the actual manifestation of class in the history of English politics and English culture since the s.

Information hiding The following is a common set of access specifiers: Only methods that are part of the same class can access private members. Protected or class-protected allows the class itself and all its subclasses to access the member. Public means that any code can access the member by its name. Although many object-oriented languages support the above access specifiers, their semantics may differ. Object-oriented design uses the access specifiers in conjunction with careful design of public method implementations to enforce class invariantsâ€”constraints on the state of the objects. A common usage of access specifiers is to separate the internal data of a class from its interface: Access specifiers do not necessarily control visibility, in that even private members may be visible to client external code. In some languages, an inaccessible but visible member may be referred to at run-time for example, by a pointer returned from a member function , but an attempt to use it by referring to the name of the member from client code will be prevented by the type checker. For example, the Java language does not allow client code that accesses the private data of a class to compile. Ruby supports instance-private and instance-protected access specifiers in lieu of class-private and class-protected, respectively. Java supports restricting access to a member within a Java package , which is the logical path of the file. However, it is a common practice when extending a Java framework to implement classes in the same package as a framework class in order to access protected members. The source file may exist in a completely different location, and may be deployed to a different. The inter-class relationship design capabilities commonly provided are compositional and hierarchical. Compositional[edit] Classes can be composed of other classes, thereby establishing a compositional relationship between the enclosing class and its embedded classes. Compositional relationship between classes is also commonly known as a has-a relationship. Therefore, a Car has an Engine. One aspect of composition is containment, which is the enclosure of component instances by the instance that has them. If an enclosing object contains component instances by value, the components and their enclosing object have a similar lifetime. If the components are contained by reference, they may not have a similar lifetime. Hierarchical[edit] Classes can be derived from one or more existing classes, thereby establishing a hierarchical relationship between the derived-from classes base classes, parent classes or superclasses and the derived class child class or subclass. The relationship of the derived class to the derived-from classes is commonly known as an is-a relationship. Therefore, a Button is a Control. Structural and behavioral members of the parent classes are inherited by the child class. Derived classes can define additional structural members data fields and behavioral members methods in addition to those that they inherit and are therefore specializations of their superclasses. Also, derived classes can override inherited methods if the language allows. Not all languages support multiple inheritance. For example, Java allows a class to implement multiple interfaces, but only inherit from one class. The hierarchy has classes as nodes and inheritance relationships as links. Classes in the same level are more likely to be associated than classes in different levels. The levels of this hierarchy are called layers or levels of abstraction. Example Simplified Objective-C 2. Definitions of subclass[edit] Main articles: Inheritance object-oriented programming , Superclass computer science , and Subclass computer science Conceptually, a superclass is a superset of its subclasses. For example, a common class hierarchy would involve GraphicObject as a superclass of Rectangle and Elipse, while Square would be a subclass of Rectangle. These are all subset relations in set theory as well, i. A common conceptual error is to mistake a part of relation with a subclass. For example, a car and truck are both kinds of vehicles and it would be appropriate to model them as subclasses of a vehicle class. However, it would be an error to model the component parts of the car as subclass relations. For example, a car is composed of an engine and body, but it would not be appropriate to model engine or body as a subclass of car. In object-oriented modeling these kinds of relations are typically modeled as object properties. In this example the Car class would have a property called parts. Object modeling languages such as UML include capabilities to model various aspects of part of and other kinds of relations. Data such as the cardinality of the objects, constraints on input and output values, etc. This information can be utilized by developer tools to generate additional code beside the basic data definitions for the objects. Things such as error checking on get and set methods. However, while some systems such as Flavors and CLOS provide a capability for more than one parent to do so at run time introduces complexity that many in the object-oriented community consider antithetical to the goals of using object classes in the first place. Understanding which class will be responsible for handling a message can get complex when dealing with more than one superclass. If used carelessly this feature can introduce some of the same system complexity and ambiguity classes were designed to avoid. For these languages, multiple inheritance may be useful for modeling but not for an implementation. However, semantic web application objects do have multiple superclasses. The volatility of the Internet requires this level of flexibility and the technology standards such as the Web Ontology Language OWL are designed to support it. A similar issue is whether or not the class hierarchy can be modified at run time. Since classes are themselves first-class objects, it is possible to have them dynamically alter their structure by sending them the appropriate messages. Semantic web objects have the capability for run time changes to classes. The rational is similar to the justification for allowing multiple superclasses, that the Internet is so dynamic and flexible that dynamic changes to the hierarchy are required to manage this volatility. Some languages, often referred to as " object-based languages ", support classes yet do not support inheritance. Examples of object-based languages include earlier versions of Visual Basic. Within object-oriented analysis[edit] Main article: Association object-oriented programming In object-oriented analysis and in UML , an association between two classes represents a collaboration between the classes or their corresponding instances. Associations have direction; for example, a bi-directional association between two classes indicates that both of the classes are aware of their relationship. For example, a "subscriber" role describes the way instances of the class "Person" participate in a "subscribes-to" association with the class "Magazine". Also, a "Magazine" has the "subscribed magazine" role in the same association. Association role multiplicity describes how many instances correspond to each instance of the other class of the association. Common multiplicities are " Abstract and concrete[edit] Main article: Abstract type In a language that supports inheritance, an abstract class, or abstract base class ABC , is a class that cannot be instantiated because it is either labeled as abstract or it simply specifies abstract methods or virtual methods. An abstract class may provide implementations of some methods, and may also specify virtual methods via signatures that are to be implemented by direct or indirect descendants of the abstract class. Before a class derived from an abstract class can be instantiated, all abstract methods of its parent classes must be implemented by some class in the derivation chain. In these languages, multiple inheritance is not allowed, but a class can implement multiple interfaces. Such a class can only contain abstract publicly accessible methods. Local and inner[edit] In some languages, classes can be declared in scopes other than the global scope. There are various types of such classes. An inner class is a class defined within another class. The relationship between an inner class and its containing class can also be treated as another type of class association. An inner class is typically neither associated with instances of the enclosing class nor instantiated along with its enclosing class. Depending on language, it may or may not be possible to refer to the class from outside the enclosing class. A related concept is inner types, also known as inner data type or nested type, which is a generalization of the concept of inner classes. This limits references to the class name to within the scope where the class is declared. Depending on the semantic rules of the language, there may be additional restrictions on local classes compared to non-local ones. One common restriction is to disallow local class methods to access local variables of the enclosing function. Metaclass Metaclasses are classes whose instances are classes. Metaclasses are often used to describe frameworks. A stand-alone class may be also designated as non-subclassable, preventing the formation of any hierarchy. Contrast this to abstract classes, which imply, encourage, and require derivation in order to be used at all. A non-subclassable class is implicitly concrete. A non-subclassable class is created by declaring the class as sealed in C or as final in Java or PHP. Typically, an executable program cannot be changed by customers. Developers can often change some classes, but typically cannot change standard or built-in ones. In Ruby , all classes are open. In Python , classes can be created at runtime, and all can be modified afterwards. Mixins[

edit] Some languages have special support for mixins , though in any language with multiple inheritance a mixin is simply a class that does not represent an is-a-type-of relationship. This section does not cite any sources. Please help improve this section by adding citations to reliable sources. Unsourced material may be challenged and removed. April Learn how and when to remove this template message In languages supporting the feature, a partial class is a class whose definition may be split into multiple pieces, within a single source-code file or across multiple files. The primary motivation for introduction of partial classes is to facilitate the implementation of code generators , such as visual designers. Using partial classes, a code generator can process a separate file or coarse-grained partial class within a file, and is thus alleviated from intricately interjecting generated code via extensive parsing, increasing compiler efficiency and eliminating the potential risk of corrupting developer code.

LANGUAGES OF CLASS pdf

4: Introduction to Computer Programming - What's a Class?

In many languages, the class name is used as the name for the class (the template itself), the name for the default constructor of the class (a subroutine that creates objects), and as the type of objects generated by instantiating the class; these distinct concepts are easily conflated.

Algonquian languages[edit] The Ojibwe language and other members of the Algonquian languages distinguish between animate and inanimate classes. Some sources argue that the distinction is between things which are powerful and things which are not. All living things, as well as sacred things and things connected to the Earth are considered powerful and belong to the animate class. Still, the assignment is somewhat arbitrary, as " raspberry " is animate, but " strawberry " is inanimate. Athabaskan languages[edit] In Navajo Southern Athabaskan nouns are classified according to their animacy, shape, and consistency. Morphologically , however, the distinctions are not expressed on the nouns themselves, but on the verbs of which the nouns are the subject or direct object. Koyukon Northern Athabaskan has a more intricate system of classification. Like Navajo, it has classificatory verb stems that classify nouns according to animacy, shape, and consistency. However, in addition to these verb stems, Koyukon verbs have what are called "gender prefixes" that further classify nouns. That is, Koyukon has two different systems that classify nouns: To illustrate, the verb stem -tonh is used for enclosed objects. When -tonh is combined with different gender prefixes, it can result in daaltonh which refers to objects enclosed in boxes or etltonh which refers to objects enclosed in bags. Australian Aboriginal languages[edit] The Dyirbal language is well known for its system of four noun classes, which tend to be divided along the following semantic lines: The Ngangikurrunggurr language has noun classes reserved for canines and hunting weapons. The Anindilyakwa language has a noun class for things that reflect light. The Diyari language distinguishes only between female and other objects. Perhaps the most noun classes in any Australian language are found in Yanyuwa , which has 16 noun classes, including nouns associated with food, trees and abstractions, in addition to separate classes for men and masculine things, women and feminine things. There are a few words with both masculine and feminine forms, generally words for relatives cousin: In names for familiar relatives, where both genders are taken into account, either the words for each gender are put together "son": Caucasian languages[edit] Some members of the Northwest Caucasian family, and almost all of the Northeast Caucasian languages , manifest noun class. Some languages have only two classes, whereas Bats has eight. The most widespread system, however, has four classes: The Andi language has a noun class reserved for insects. In all Caucasian languages that manifest class, it is not marked on the noun itself but on the dependent verbs, adjectives, pronouns and prepositions. Nigerâ€”Congo languages[edit] Nigerâ€”Congo languages can have ten or more noun classes, defined according to non-sexual criteria. Certain nominal classes are reserved for humans. The Fula language has about 26 noun classes exact number varies slightly by dialect. According to Steven Pinker , the Kivunjo language has 16 noun classes including classes for precise locations and for general locales, classes for clusters or pairs of objects and classes for the objects that come in pairs or clusters, and classes for abstract qualities. While no single language is known to express all of them, most of them have at least 10 noun classes. Additionally, there are polyplural noun classes. A polyplural noun class is a plural class for more than one singular class. Classes 6 and 10 are inherited as polyplural classes by most surviving Bantu languages, but many languages have developed new polyplural classes that are not widely shared by other languages. Specialists in Bantu emphasize that there is a clear difference between genders such as known from Afro-Asiatic and Indo-European and nominal classes such as known from Nigerâ€”Congo. Languages with nominal classes divide nouns formally on the base of hyperonymic meanings. The category of nominal class replaces not only the category of gender, but also the categories of number and case. This seems to them to be inconsistent with the way other languages are traditionally considered, where number is orthogonal to gender according to the critics, a Meinhof-style analysis would give Ancient Greek 9 genders. If one follows broader linguistic tradition and counts singular and plural as belonging to the same class, then Swahili has 8 or 9 noun classes, Sotho has 11 and Ganda has The Meinhof numbering tends to be used in scientific works dealing with comparisons of different Bantu languages. For this reason, noun classes are often referred to by combining their singular and plural forms, e. However not all Bantu languages have these exceptions. In Ganda each singular class has a corresponding plural class apart from one class which has no singularâ€”plural distinction; also some plural classes correspond to more than one singular class and there are no exceptions as there are in Swahili. For this reason Ganda linguists use the orthogonal numbering system when discussing Ganda grammar other than in the context of Bantu comparative linguistics , giving the 10 traditional noun classes of that language. Italian , for example, has a group of nouns deriving from Latin neuter nouns that acts as masculine in the singular but feminine in the plural: These nouns are still placed in a neuter gender of their own by some grammarians. Here is a complete list of nominal classes in Swahili:

5: Language Classes | ISL International School of Languages

Language and Social Class 2 4 Variables of Social Class â€¢Power - The degree to which a person can control other people â€¢ Wealth - Objects or symbols owned by people which have.

6: Courses | Department of Linguistics

Bonjour, Hello, Hola, Bom Dia, Buongiorno! On this site I have complied over activities and teaching suggestions from my 20 years of teaching world language.

7: Noun class | language | www.enganchecubano.com

About Class Central Class Central is a search engine and reviews site for free online courses popularly known as MOOCs or Massive Open Online Courses. MOOC Report.

8: Global Studies and Languages | MIT OpenCourseWare | Free Online Course Materials

Study free online Programming Languages courses & MOOCs from top universities and colleges. Read reviews to decide if a class is right for you. Read reviews to decide if a class is right for you. Follow to get an email when new courses are available Follow.

9: å…¨ç¾Žä¸å°•å¸ä¸æ–‡æ•™å¸ˆå••ä¼š| Chinese Language Association of Secondary-Elementary Scho

The Language class in CodeIgniter is meant to provide an easy and lightweight way to support multiplelanguages in your application. It is not meant to be a full implementation of what is commonly called internationalization and localization.

LANGUAGES OF CLASS pdf

Bucks County, Pennsylvania, Orphanse Court Records 1852-1900 Unity and complexity Towards the 5th phase school improvement : future directions for the field Janet Chrispeels and Alma Harr How to finance your company Bailin love gauge field theory List of noble gases Mammals of the canyon country Selected short fiction Illustrated Flora of Keoladeo National Park, Bharatpur, Rajasthan The Breakdown of Democratic Party Organization, 1940-1980 16. Underwear Spy 209 8. Evolution and genetics Jill Bailey Directory of exhibition spaces Egyptian honeymoon Stunt Kite Basics Canadas Federal System Being Treatise On Canadian Constitutional Law Under the British North America Act Aptitude tutorial point Statements by Orthodox, Conservative and Reform rabbis on the architecture of the synagogue. DeNoizr. Productivity Booster for the Common Man Putting Medicare consumers in charge Building beehives John legend all of me piano Lord, lift me up and let me stand Bearing witness to the call StressAlyzer CD-ROM (Stand-Alone Version) Easy way to learn autocad 2007 Miss Bindergarten Stays Home From Kindergarten (Miss Bindergarten Books) A systemoptimization perspective for supply chain network integration All the Presidents Women Comparison of the intelligence and training of school children in a Massachusetts town Statues Without Shadows The Irish Policeman, 1822-1922 Saxon, medieval and post-medieval settlement at Sol Central, Marefare, Northampton Families, work, and housework Russia, Eurasian States, and Eastern Europe 1999 (Russia Eurasian States and Eastern Europe 1999) Sslc hindi notes 2017 18 Visual Encyclopedia of Animals A song of fire and ice series The literary dream in French romanticism Fairy Stories Do Sometimes Come True