1. ¿Cuál de los siguientes términos se usan en el DTD?
2. Name.
3. Notation.
4. Type.
5. Use.
6. ¿Para qué sirve un DTD?
7. Para crear un XML, puede estar en un archivo aparte o no.
8. Para validar un archivo XML, el DTD tiene crearse en un archivo aparte.
9. Para validar un archivo XML.
10. Para proporcionar a elementos y atributos nombres únicos.
11. ¿Cuál de los siguientes DTD está bien formado? Teniendo en cuenta que son completos.
12. <DOCTYPE nombre[

<!ELEMENT nombre(#PCDATA)>

]>

1. <!DOCTYPE nombre[

<!ELEMENT nombre ANY

]>

1. <!DOCTYPE nombre[

<!ELEMENT nombre(otroElemento)>

]>

1. <!DOCTYPE nombre[

<!ELEMENT nombre(nombre|hijo1)>

]>

1. ¿Cuál de los siguientes operadores sirve para que haya de 0 a 1 ocurrencias de ese elemento en la declaración de un DTD?
2. +
3. \*
4. ?
5. minOccurs=”0” maxOccurs=”1”
6. ¿Cuál de los siguientes elementos puede tener los elementos a, b como hijos colocados sin ningún orden concreto y que pueda haber varios de cada?
7. <!ELEMENT elemento (a | b)>
8. <!ELEMENT elemento (a, b)>
9. <!ELEMENT elemento ((a|b)|(b|a))>
10. <!ELEMENT elemento (a|b)\*>
11. ¿Cuál de los siguientes elementos tiene de hijos a los elementos a, b y c, siendo c opcional y que solo haya a o b?
12. <!ELEMENT elemento (a|b, c?)>
13. <!ELEMENT elemento (a | b,c)\*>
14. <!ELEMENT elemento ((a|b)|c)>
15. <!ELEMENT elemento (a, b, c)>
16. ¿Para un atributo declarado en un DTD, que valores puede tener el tipo CDATA?
17. Puede tener cualquier tipo de carácter.
18. Contendrá una cadena de caracteres de cualquier tipo.
19. Puede contener una cadena de todos los caracteres excepto >, <, &, ‘ y”.
20. Contendrá un valor único en todo el documento y será una cadena de caracteres.
21. ¿Cuál de los siguientes tipos de atributos puede contener varios valores de cadenas de caracteres validas XML, es decir solo letras, números, guiones y puntos?
22. NMTOKEN
23. CDATA
24. NOTATION
25. NMTOKENS
26. ¿Cómo afecta a los atributos el indicador #REQUIRED?
27. Hace que el atributo tenga un valor fijo.
28. El atributo debe aparecer en el documento XML, pero puede estar vacío.
29. El atributo debe aparecer en el documento XML, pero tiene que tener valor.
30. Si el atributo tiene un valor diferente al especificado en el DTD, da error.
31. ¿Dónde se declaran los XSD?
32. En el propio documento XML a validar o en un archivo a parte.
33. En un documento a parte, cuya extensión es xs.
34. Se enlaza el archivo con un espacio de nombres en la raíz a un archivo XSD.
35. Un archivo XSD donde se declara cual es el archivo a validar.
36. ¿Las declaraciones de atributos de un XSD, de que tipo son?
37. Siempre son simples.
38. A veces simples, otras complejos.
39. Siempre son complejos.
40. Tiene un tipo aparte.
41. ¿Cómo se valida un documento XML con un documento XSD?
42. xsd=”nombre.xsd”
43. Se tiene que hacer el XSD en el propio documento, no en otro aparte.
44. xs:schema=”nombre.xsd”
45. xmlns:xsi=”URI” xsi:noNameSpaceSchemaLocation=”nombre.xsd”
46. ¿Cómo se declara un elemento simple en XSD?
47. <element>”nombreDelElemento”</element>
48. <element name=”nombreDelElemento”/>
49. <element name=”nombreDelElemento” use=”Obligatorio o no”/>
50. <!ELEMENT name=”nombreDelElemento”>
51. ¿Cómo se declara un elemento complejo en XSD?
52. Dentro de una etiqueta de element, usando otras etiquetas como complexType y sequence.
53. Creando un elemento simple y después en una etiqueta complexType le haces referencia con el atributo ref.
54. No se puede.
55. Se declara con el atributo type en un elemento simple.
56. ¿Qué hay de diferencia entre un elemento simple y uno complejo?
57. Los tipos que pueden tener
58. Donde se declaran
59. El elemento simple no puede tener hijos, el complejo sí.
60. No hay diferencia alguna, pero se usan para entender mejor el XSD.
61. ¿Qué hace la etiqueta sequence?
62. Dentro se hacen las declaraciones de los hijos de la etiqueta en la que se declara, aunque sea simple o compleja.
63. Es donde se declaran los hijos del elemento complejo, se validan en el orden establecido dentro de la etiqueta.
64. Sirve para declarar en que secuencia irán los atributos del elemento.
65. Ninguna de las anteriores es correcta.
66. ¿Cuál de los siguientes NO es un modelo de construcción de los enseñados en clase?
67. Modelo anidado/de muñecas rusas.
68. Diseño Plano.
69. Modelo de árbol simple.
70. Diseño con tipos con nombres reutilizables.
71. ¿Cómo se establecen que hijos tendrá una etiqueta en concreto?
72. Esa etiqueta tiene que ser compleja y se declaran nuevas dentro de una sub etiqueta que puede ser sequence.
73. Se hacen referencia o se declaran dentro del propio elemento, sea simple o complejo. Dentro de una sub etiqueta que no es sequence
74. Se hacen referencia o se declaran dentro del propio elemento, que tiene que ser simple. Dentro de una sub etiqueta que no es sequence
75. Se hacen referencia o se declaran dentro del propio elemento, que tiene que ser complejo. Dentro de una sub etiqueta que puede ser sequence
76. ¿Qué son MinOccurs y MaxOccurs ?
77. Son metadatos para indicar la cardinalidad.
78. Son elementos para indicar la cardinalidad.
79. Son atributos que indican la cardinalidad.
80. Es vocabulario especial del XSD que se usa en los comentarios.
81. ¿A que equivale un MinOccurs=”1”, MaxOccurs=”unbounded” en un DTD?
82. \*
83. +
84. Está mal utilizados ya que no son atributos.
85. Ninguna es correcta.
86. ¿Cómo declaro que una elemento pueda aparecer o no en un XSD?
87. Con el atributo use y asignándole required
88. Con el atributo use y asignándole prohibited
89. Con minOccurs=”1” y ya.
90. Ninguna es correcta.
91. ¿Para qué sirve crear un tipo nuevo en un XSD?
92. Restringe los valores de una forma concreta.
93. Se crean a partir de tipos base ya definidos.
94. Puedes crear tipos nuevos a partir de otros tipos nuevos, añadiéndole más posibilidades.
95. Todas son correctas.
96. ¿Cuál de estas expresiones regulares valida las cadenas “abc897” y “khf654” pero no “ABC”?
97. [a-z]{3}\d{3}
98. [a-Z]{3}\d{3}
99. [A-Z]{3}\D{3}
100. Ninguna es correcta.
101. ¿Dónde se utiliza y para qué sirve la etiqueta union?
102. En elementos complejos, para unir diferentes elementos hijo.
103. En elementos complejos, se utiliza como puerta lógica “OR”
104. En tipos complejos, para crear un tipo a partir de otros tipos.
105. En tipos simples, para crear un tipo a partir de otros tipos.
106. ¿Dónde se utiliza y para qué sirve la etiqueta choice?
107. En elementos complejos, sirve como alternativa de los diferentes hijos para que solo puede aparecer uno cuantas veces sea establecido.
108. En elementos complejos, sirve como alternativa de los diferentes hijos para que solo puede aparecer uno cuantas veces sea establecido.
109. En tipos simples, sirve como alternativa de los diferentes tipos para que solo puede usar uno.
110. En tipos complejos, sirve como alternativa de los diferentes tipos para que solo puede usar uno.