NSML – A Markup Language for News Stories
Version 3.0

1 Introduction
The purpose of this document is to define the News Story Markup Language (NSML). NSML is an SGML based markup which is used to express all the content and information about a story. It also can describe a form that can be used to describe the presentation of fields when displayed.

NSML is used to express seven separate aspects of a story. NSML expresses meta information about a story in the <head> section; it expresses the appearance of the story when displayed in the <form> section; it expresses the story content in the <fields>, <body>, and <aeset> sections; and it expresses arbitrary attachments in the <field-atts> and <aeset-atts> sections.

1.1 Changes from 2.7 to 2.8
The <password> tag was added.
Document the limitations on the idref of an <a/> tag and the id of an <ae> tag.
Note that the sequence of anchored elements conform to the sequence of anchors within the <body> in the <ae> tag description.
The fields ITEM-CHANNEL, MOS-DURATION, and RUNS-TIME added to list of reserved fields in section 7.1.
The example was replaced in section 12.

1.2 Changes from 2.8 to 3.0
NSML is now well formed XML. NSML documents can be validated against the NSML XML schema. As a well formed XML document, tag names are case sensitive.
The <nsml version> attribute is now required and it is now simply a number.
The <look> tag has been removed.
The <story> tag has been removed.
The <form>, <fields>, <body>, and <aeset> tags are now children of the <nsml> tag.
The <img> tag has been removed.
The <meta deleted> attribute has been removed.
Two new attributes were added to the <meta/> tag, “wordlength” and “version”.
The <cell> tag was replaced by the <stringcell>, <comboboxcell>,<checkboxcell>, <datecell>, and <durationcell> tags. All of these tags were added as content to the <row> tag.
A “wg” attribute was added to all cell tag types.
A “limit” attribute was added to all cell tag types to restrict the content of the field to its edit size.
The <label>, <entry>, <group>, and <list> tags were added as part of the <comboboxcell> content.
The <f> tag was replaced by the <string>, <boolean>, <date>, and <duration> tags. All of these tags were added as content to the <fields> tag.
A “bg” attribute was added to the <string>, <boolean>, <date>, and <duration> tags.
The `<bg>` tag was added to allow background color highlighting to a sequence of characters. It is similar to the `<b>`, `<i>`, and `<u>` tags.

The `<rtl>` tag was added to denote a right to left sequence of characters. It is similar to the `<b>`, `<i>`, and `<u>` tags.

Font, family, and pitch attributes were added to the `<body>`, `<p>`, `<aeset>`, `<ae>`, `<mc>`, and `<ap>` tags.

A `<bg>` tag was added to hold a background color for text.

The `<link>` tag was added as content to the `<p>` and `<ap>` tags. The `<show>` and `<url>` tags were added as content of the new `<link>` tag.

The `<egroup>` tag was added as content of the `<head>` tag.

The `<storygroupid>` tag was added as content of the `<head>` tag.

The `<form>` and `<row>` tags no longer set default values for all of the cell tag attributes.

Empty tags are now denoted with the “/” sequence ( `<a/>`, `<meta/>`, `<pb/>`, `<tab/>`, `<wp/>`, etc.)

All tag attributes names are in lower case.

All tag attribute values are enclosed in quotes.

All tag attributes have the form: name="value"

End tags are no longer optional. Neither are any start tags.

Hidden text has been eliminated. Most of the content was converted to attributes added the `<ae>` and `<mc>` tags. A `<hidden>` tag was added to the `<ae>` content to accommodate text unique to ‘J’ type anchored elements.

### 1.3 Precedence

Where the descriptive text of this specification differs with the syntax of the NSML XML Schema, the Schema takes precedence and should be considered the correct definition of NSML.

### 2 NSML Grammar

The elements of a story are identified with NSML tags. These tags are organized in a hierarchy. At the top of the hierarchy is the `<nsml>` tag which delimits a complete NSML document and contains the story meta information, story content, story presentation, and attachment elements. The other elements of a story are nested inside these elements as described by the NSML grammar.

The story grammar defines the ordering and nesting of the tags and story elements. There are two sets of language elements: terminal – the actual words in the NSML language – and non-terminal – everything else. The terminals are formatted as tags with their enclosing brackets “<>”, non-terminals are displayed in italic. There is one predefined non-terminal, `plain_text`. `Plain_text` is defined to be regular characters, and character entities denoted by the ampersand character. A CDATA section can be used to encapsulate `plain_text` that contains many NSML markup characters that would otherwise be converted into character entities.

Elements followed by a ‘?’ are optional and occur at most one time.

Elements followed by a ‘*’ are repeatable zero or more times.
Elements must appear in the order they are listed. Elements that are listed as a group of elements separated by a '|' indicate a choice. When the group is defined to occur more than one time, the resulting set of elements can occur in any order.

```
nsmi_tag ::= <nsml>
  </nsml>

head_tag ::= <head>
  </head>

form_tag ::= <form> row_tag * </form>

field_atts_tag ::= <field-atts> attachment_tag * </field-atts>

aset_atts_tag ::= <aaset-atts> attachment_tag * </aaset-atts>

row_tag ::= <row> row_content * </row>


label_tag ::= <label> plain_text? </label>

comboboxcell_content ::= entry_tag | <group> entry_tag * </group> | <list> entry_tag * </list>

entry_tag ::= <entry> plain_text? </entry>

fields_tag ::= <fields> fields_content * </fields>```
<table>
<thead>
<tr>
<th>fields_content</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;string&gt; plain_text ? &lt;string&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>boolean_value</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= “true”</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>date_value</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= a decimal number representing seconds since 00:00:00 UTC, January 1, 1970</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>duration_value</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= a decimal number representing seconds</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>body_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;body&gt; p_tag * &lt;/body&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>p_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;p&gt; p_content * &lt;/p&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>p_content</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= content_tag</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>content_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;pi&gt; text_content * &lt;/pi&gt;</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>text_content</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= style_tag</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>text_and_links</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;link&gt; show_tag ? url_tag &lt;link&gt;</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>show_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;show&gt; plain_text ? &lt;/show&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>url_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;url&gt; plain_text ? &lt;/url&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>style_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;b&gt; text_content * &lt;/b&gt;</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>aestet_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;aestet&gt; ae_tag * &lt;/aestet&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ae_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;ae&gt; hidden_tag ? ae_content * &lt;/ae&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>hidden_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;hidden&gt; plain_text ? &lt;/hidden&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ae_content</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= mc_tag</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>mc_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;mc&gt; ap_tag * &lt;/mc&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ap_tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>::= &lt;ap&gt; ap_content * &lt;/ap&gt;</td>
</tr>
</tbody>
</table>
3 NSML Basics

NSML defines a set of tags that are embedded in the story document. NSML tags are delimited by a set of angle brackets (< and >) and contain a tag name and optional tag attributes. For example, the tag `<string id="title">` has a tag name of `string` and one attribute `id="title"`. Each tag has its own set of attributes that may be applied to it. In the previous example, the attribute name `id` is set to the attribute value `title`. In general, attributes are optional and defaults are assumed if they are not specified. There are some exceptions to this. The description of each tag in this document points out all of the required attributes.

3.1 Start and End tags

Most tags define and affect a specific part of an NSML document. This part of the document begins where the tag with its attributes (known as the start tag) appears in the document and continues to the corresponding end tag. The end tag has the same name as its corresponding start tag but starts with a `/` sequence rather than just a `<` sequence. For example, `</string>` is the `<string>` tag’s end tag. End tags have no attributes. The part of the document enclosed in a set of start and end tags is referred to as the tag’s content.

The number of characters in a tag, including the tag name and all attribute names and values but not including the tag delimiters, cannot exceed 500. This excludes comments and CDATA sections (described below).

3.2 Empty tags

Some tags never have any content. They may have attributes. The format of these empty tags are similar to start tags. The only difference is that empty tags end with the two character sequence “/>”. Some examples are `<a/>`, `<tab/>`, `<pb/>`, and `<wp/>`.

3.3 Character set

The set of characters allowed in tag names and attribute names is the ASCII characters a-z, 0-9, dot and dash. The first character of a name is always in the set a-z. A name may have a maximum of 12 characters.

Attribute values can be divided into three categories, TEXT, TOKEN and NUMBER values. A NUMBER value is a string of decimal digits from the ASCII set 0-9. A TOKEN value is a string of ASCII characters which has the same lexical constraints as tag and attribute names (same character set, same restrictions,
case-insensitive, etc.) A TEXT value is a string of bytes in the range 0x20 to 0xff excluding the double quote character. TEXT values are case-sensitive. If a TEXT value contains characters not allowed in a TOKEN then it must be enclosed in double quotes. Values of type TEXT are limited to 100 bytes.

The characters allowed in plain_text content are undefined. The plain_text content is treated as a sequence of bytes. There are only two exceptions to this. When the plain_text content is parsed byte values in the range 0x00 to 0x1f are stripped from the content and character entity conversion is done on the plain_text content. The characters in the range 0x00 to 0x1f may be included in the plain_text content if represented as character entities.

Plain_text can be encapsulated in a CDATA section. CDATA sections are used to escape blocks of text containing characters that would otherwise be recognized as NSML markup. A CDATA section starts with the nine character sequence ( <!--[CDATA[) and ends with the three character sequence ( ]]>). Within a CDATA section, only the end sequence ( ]]>) is recognized as markup, so that left and right angle brackets and ampersands may occur in the literal form; they need not be escaped using character entities. CDATA sections cannot nest. Byte values in the range 0x00 to 0x1f, except 0x09, 0x0A, and 0x0D will be stripped from CDATA sections.

### 3.4 Character Entities

Anywhere in plain_text characters may be replaced with character entities. This is useful when the plain_text contains NSML markup characters that could confuse an NSML parser. NSML recognizes the full set of numeric character entities. Numeric character entities are 1 to 3 decimal digits preceded by &# and followed by a semicolon, ';'. Leading zeros are ignored so &#062; is the same as &#62;. The named character entities that may appear in plain_text are identified in table 2.1.

<table>
<thead>
<tr>
<th>Named Entity</th>
<th>Numeric Entity</th>
<th>symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&gt;</td>
<td>&gt;</td>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt;</td>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&amp;</td>
<td>&amp;</td>
<td>&amp;</td>
<td>Ampersand</td>
</tr>
</tbody>
</table>

### 3.5 Comments

Comments may be included anywhere (outside of tags) within an NSML document. Comments can only be seen when viewing the raw form of the document. In almost all cases comments should be hidden from the user’s view. It is allowable for applications that process a document to strip the comments from the processed document. A comment is any characters between the four character start, "<!--" and the four character end "-->", comment markers (i.e. <!-- Comment Text Here -->). No translation of comment text is performed. This means that the characters '&', '<', and '>' will be stored in the NSML document unchanged. Comments cannot be nested - the first "-->", encountered terminates the comment.

For details on other text that is not intended for display, see the <hidden> tag.

### 3.6 Base Units

Many of the attribute values are measurements expressed in base units. This measurement is equal to one-fourth of the average character width of the “system font”. This unit is based on the Microsoft Windows definition of its Dialog Base Unit. This measurement is device independent, so that applications can use a single measurement to create a similar appearance on different types of display devices or different font sizes.
4 NSML top level elements

4.1 The <nsml> tag

\[<\text{nsml}>\]

Function:
Delimits a complete NSML document.

Attributes:
Version [REQUIRED] decimal number

Contains:
\text{head} \ tag \ ?
\text{form} \ tag \ ?
\text{fields} \ tag \ ?
\text{body} \ tag \ ?
\text{aest} \ tag \ ?
\text{field \ atts} \ tag \ ?
\text{aest \ atts} \ tag \ ?

Used in:
Nothing

The version attribute is optional. Its value is TEXT that defines the NSML version used to compose a document. It must read: version="3"

If not specified an NSML parser is to assume that the document is an earlier version of NSML.

4.2 The <head> tag – define information about the story

\[<\text{head}>\]

Function:
Defines information about a story or attached to a story that is not strictly content.

Attributes:
None

Contains:
\text{meta} \ tag \ ?
\text{rgroup} \ tag \ ?
\text{wgroup} \ tag \ ?
\text{egroup} \ tag \ ?
\text{wcode} \ tag \ ?
\text{dist} \ tag \ ?
\text{source} \ tag \ ?
\text{formname} \ tag \ ?
\text{storyid} \ tag \ ?
\text{storygroupid} \ tag \ ?
\text{password} \ tag \ ?

Used in:
\text{nsml} \ tag

Comment: Note that these have been promoted due to the elimination of the story tag.

Comment: This is needed to better distinguish between NSML versions. A iNEWS NSML parser will assume version 2 if the <nsml> tag does not have the required version attribute value.

Comment: Editorial groups
### 4.3 The `<form>` tag – define the appearance of the story fields

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th>Contains</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;form&gt;</code></td>
<td>Defines a presentation/appearance of the story’s fields. Descendents inherit all attributes unless overridden.</td>
<td>None</td>
<td><code>row tag *</code></td>
<td><code>nsml_tag</code></td>
</tr>
</tbody>
</table>

The `<form>` and `</form>` tags enclose a presentation description for the fields of a story. It does not contain any story content. All story content is defined within the `<fields>`, `<body>`, and `<aeset>` tags. A form is a definition of the layout of the story fields on a “page” when viewed. An NSML document need not contain a form definition. The presentation of the story is the responsibility of the application displaying the document that must acquire a form.

### 4.4 The `<fields>` tag

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th>Contains</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;fields&gt;</code></td>
<td>Delimits the story fields content.</td>
<td>None</td>
<td>`(string_tag</td>
<td>boolean_tag</td>
</tr>
</tbody>
</table>

The new field types.

### The `<body>` tag

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th>Contains</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;body&gt;</code></td>
<td>Defines the story text content.</td>
<td></td>
<td><code>p_tag</code> optional</td>
<td><code>nsml_tag</code></td>
</tr>
</tbody>
</table>
The `<body>` and `</body>` tags enclose the text of the story. All text must be within paragraphs in the body element.

The `tabs` attribute is optional. Its value is a list of *decimal numbers* that, if specified, defines the tab stop positions for all `<tabs/>` within the text. Each number in the list is the distance from the last stop. The last value in the list may be a dash, this indicates that the last number in the list is to repeat indefinitely. The numbers are in base units. For example, `tabs="40 -"` may be specified to place stops at every 40 base units. If not specified `tabs="24"` is assumed. If more than one number is specified in the list, the numbers are separated by spaces (i.e. `tabs="72 144"`).

The `script=` and `width=` attributes are optional. They are used to preserve the margin settings used for word wrapping by the last application that modified the story. The `width` value is a decimal number that specifies the width, in base units, used to word wrap the text contained in the `<body>` tag. The `script` value is a decimal number that specifies the width, in base units, used to word wrap the text in the anchored elements. If the `script` attribute is not present this indicates the story is not scripted. If the `width` attribute is not present the application displaying the body text must choose an appropriate width. See description of related `<wp/>` tag.

The `pindent=` attribute is optional. Its value is a *decimal number* that defines the left paragraph indent for all paragraphs in the body of the story. The indent value is expressed in base units from the left. If not set it is assumed to be zero.

The `rindent=` attribute is optional. Its value is a *decimal number* that defines the right paragraph indent for all paragraphs in the body of the story. The indent value is expressed in base units from the right. If not set it is assumed to be zero.

The `findent=` attribute is optional. Its value is a *decimal number* that defines the indent for the first line of all paragraphs in the body of the story. The indent value is expressed in base units from the left. If not set it is assumed to be the same as `pindent`.

The `font=` attribute is optional. Its value is a string that is the name of a font to be used for display throughout the `<body>` of the story.

The `family=` attribute is optional. Its value is a *decimal number* in the range 0 to 5 that represents a font-family. If not set it is assumed to be zero. The meaning of the font-family values are:

- 0  Don’t care
- 1  Roman
- 2  Swiss
- 3  Modern
- 4  Script
- 5  Decorative

The `pitch=` attribute is optional. Its value is a *decimal number* in the range 0 to 2 that represents a pitch setting. If not set it is assumed to be zero. The meaning of the pitch values are:

- 0  default
- 1  fixed
- 2  variable
4.5 The `<aeset>` tag

```xml
<aeset>
    Function:
    Defines the set of anchored elements within the story.
    Attributes:
    - `font`  string
    - `family`  decimal number (0 – 5)
    - `pitch`  decimal number (0 – 2)
    Contains:
    - `ae_tag` *
    Used in:
    - `nsml_tag`
```

See the description of the attributes under the `<body>` tag description.

4.6 The `<field-atts>` tag – Defining field attachments

```xml
<field-atts>
    Function:
    Delimits the field attachments.
    Attributes:
    None
    Contains:
    - `attachment_tag` *
    Used in:
    - `nsml_tag`
```

4.7 The `<aeset-atts>` tag – Defining anchored element set attachments

```xml
<aeset-atts>
    Function:
    Delimits the anchored element set attachments.
    Attributes:
    None
    Contains:
    - `attachment_tag` *
    Used in:
    - `nsml_tag`
```
5 Story Header - the <head> tag content

5.1 The <meta/> tag – Define story meta information

| Function: | Defines meta information about a story. |
| Attributes: | |
| wire | “f” | “b” | “u” | “r” | “o” |
| mail | “read” | “unread” |
| locked | “pass” | “user” |
| words | decimal number |
| rate | decimal number |
| break | boolean value |
| mcserror | boolean value |
| hold | boolean value |
| float | boolean value |
| wordlength | decimal number |
| version | decimal number |

Contains: Nothing

Used in: <head_tag>

The `wire` attribute is optional. If present it indicates that the story is an original, unmodified story received from a wire service provider. Its value must be one of “f” (flash), “b” (bulletin), “u” (urgent), “r” (routine), or “o” (other) to indicate the priority of the wire story. If not present, the story is not a wire story.

The `mail` attribute is optional. If present it indicates that the story is an original, unmodified story received as a mail message. Its value must be one of “read” or “unread” to indicate whether the receiver of the mail message has viewed it. If not present, the story is not a mail message.

The `locked` attribute is optional. If present it indicates that a user has locked the story. A user may restrict access to a story by locking it in one of two ways. Its value must be one of “pass” or “user”. If the value is “pass” then access is granted to users that can supply the correct password. If the value is “user” then access is granted only to the user who locked the story. If not present, the story is not locked. See the description of the `<password>` tag.

The `words` attribute is optional. Its value is a decimal number that, if present, specifies the number of countable words in the body of the story. Countable words are those words that contribute to the audio read time of a story. If not present, the number of words is assumed to be zero.

The `rate` attribute is optional. Its value is a decimal number that, if present, specifies the read rate to be used to calculate the audio read time. The read rate is in words per minute. If not present, the rate is assumed to be zero.

The `break` attribute is optional. Its value is either either “true” / “1” or “false” / “0”. A true value indicates that the story is a story in a rundown which marks a break or divider between segments of the rundown. If this attribute is not present, false is assumed.
The `mcserror=` attribute is optional. Its value is either "true" / "1" or "false" / "0". A true value indicates that the story is a story in a rundown which contains machine control instructions and at least one of the instructions cannot be understood by the machine control subsystem. If this attribute is not present, false is assumed.

The `hold=` attribute is optional. Its value is either either "true" / "1" or "false" / "0". A true value indicates that the story is not eligible for automatic purge. A user will place a hold on a story that he wants to protect from the automatic database story purge policy. If this attribute is not present, false is assumed.

The `float=` attribute is optional. Its value is either either "true" / "1" or "false" / "0". A true value indicates that the story is a story in a rundown which has been "floated" by a user. A floating story deserves special treatment in the rundown because it is not yet included as a story to go on-air. If this attribute is not present, false is assumed.

The `wordlength=` attribute is optional. Its value is a *decimal number* that, if present, specifies the number of characters that constitute a word in the body of the story.

The `version=` attribute is optional. Its value is a *decimal number* that, if present, specifies the version number of the story. This number starts at 0 when a story is created.

### 5.2 The `<rgroup>` Tag

```
<rgroup>
    Function: Defines the name of the read group assigned to the story.
    Attributes: number  *decimal number*
    Contains:         plain_text ?
    Used in:         head_tag
```

The `number=` attribute is optional. It is a *decimal number* which is the internal group number associated with the group name. It is valid to omit the group name and just specify the group number as in `<rgroup number="20"/>`. If not specified the story has no read group assigned to it.

### 5.3 The `<wgroup>` Tag

```
<wgroup>
    Function: Defines the name of the write group assigned to the story.
    Attributes: number  *decimal number*
    Contains:         plain_text ?
    Used in:         head_tag
```

The `number=` attribute is optional. It is a *decimal number* which is the internal group number associated with the group name. It is valid to omit the group name and just specify the group number as in `<wgroup number="20"/>`. If not specified the story has no write group assigned to it.
5.4 The <egroup> tag

Function:
Defines the name of the editorial group assigned to the story.

Attributes:
- number  decimal number

Contains:
- plain_text

Used in:
- head_tag

The `number` attribute is optional. It is a `decimal number` which is the internal group number associated with the group name. It is valid to omit the group name and just specify the group number as in `<egroup number="20"/>`. If not specified the story has no editorial group assigned to it.

Comment: Editorial groups.

5.5 The <wcode> tag

Function:
Defines the wire distribution codes assigned to the story when it was originally received.
This is only relevant if the story originated from a wire service provider.
Not all wire stories will have a wire distribution code.

Attributes:
- None

Contains:
- plain_text

Used in:
- head_tag

5.6 The <dist> tag

Function:
Defines the distribution code that a user has attached to the story.

Attributes:
- None

Contains:
- plain_text

Used in:
- head_tag
5.7 The `<source>` tag

```
<source>
Function:
Defines the name of the News DataBase Server that generated the story.
Attributes:
None
Contains:
plain_text?
Used in:
head_tag
```

5.8 The `<formname>` tag

```
<formname>
Function:
Defines the form that was used to create the first instance of the story.
The form also determines which fields must be preserved when the story is edited.
Attributes:
None
Contains:
plain_text?
Used in:
head_tag
```

5.9 The `<storyid>` tag

```
<storyid>
Function:
Defines the identification string used by the News DataBase Server to locate a story.
The format and content of the `<storyid>` is not specified in NSML.
Attributes:
None
Contains:
plain_text?
Used in:
head_tag
```

5.10 The `<storygroupid>` tag

```
<storygroupid>
Function:
Defines the string used by the News DataBase Server to identify a group of stories.
Each story in the group will have this same identifier.
The format and content of the `<storygroupid>` is not specified in NSML.
Attributes:
None
Contains:
plain_text?
Used in:
head_tag
```
5.11 The <password> tag

```plaintext
<password/>
```

**Function:**
Defines the password assigned to the story if it is locked with a password. This should only be present if the `locked=“pass”` attribute is present in the `<meta/>` tag. This will only be included in documents sent and received by iNEWS 2.5 and above systems.

**Attributes:**
None

**Contains:**
`plain_text`

**Used in:**
`head_tag`

6 Story Presentation – the <form> tag content

6.1 The <row> tag

```plaintext
<row/>
```

**Function:**
Defines a presentation/appearance of a row of fields.

**Attributes:**
None

**Contains:**
`stringcell_tag | checkboxcell_tag | comboboxcell_tag | datecell_tag | durationcell_tag`

**Used in:**
`form_tag`

6.2 The <stringcell> tag

```plaintext
<stringcell/>
```

**Function:**
Defines a presentation/appearance of a field.

**Attributes:**
- `idref`: id of a tag in the `<fields>` section
- `length`: decimal number
- `style`: "b" | "i" | "u" | "" | space separated list of those letters
- `align`: "left" | "right" | "center"
- `ro`: boolean_value
- `aready`: boolean_value
- `llength`: decimal number
- `llstyle`: "b" | "i" | "u" | "" | space separated list of those letters
- `lalign`: "left" | "right" | "center"
- `lposition`: "left" | "right" | "top" | "bottom"
- `limit`: boolean_value
- `wg`: decimal number
- `mandatory`: boolean_value

**Contains:**
`plain_text`

**Used in:**
`row_tag`

**Comment:** Converted to an attribute / value pair instead of each value being a “compact” attribute.

**Comment:** More former “compact” attributes.
The \textit{idref} attribute is optional. It identifies the specific field from the \textit{<fields>} section which is to be displayed in this cell. It is the \textit{id} value that is matched. If not specified the cell is assumed to be blank and read-only.

The \textit{length} attribute is optional. Its value is a \textit{decimal number} that, if present, defines the length in base units of the cell. If not specified the length of the cell is assumed to be zero.

The \textit{style} attribute is optional. It defines the physical style of the field content displayed in the cell. If not specified the cell inherits a style from the enclosing \textit{<row>} or \textit{<form>} tags. If no style is specified in either the \textit{<cell>}, \textit{<form>} or \textit{<row>} then the style defaults to normal. The style value must be one of “\textit{b}” (bold), “\textit{i}” (italic), or “\textit{u}” (underline) or a space separated list of those characters. For example \textit{style=”b u”} specifies a style of bold-underline. To specify a normal style, \textit{style=’’} may be used.

The \textit{align} attribute is optional. It defines the alignment of the field content displayed in the cell. The value must be one of “\textit{left}”, “\textit{right}”, or “\textit{center}”. If not specified, “\textit{left}” alignment is assumed.

A cell that references a field can inherit some behavior that is associated with the field’s \textit{id}. For example, a cell that references the TOTAL-TIME field will be read-only and must contain the sum of the TAPE-TIME field and the story’s AUDIO-TIME field. This behavior is an implied attribute of the field with \textit{id=TOTAL-TIME}.

The \textit{ro} attribute is optional. Its value is either “\textit{true}” / “\textit{1}” or “\textit{false}” / “\textit{0}”. A true value indicates that the cell is \textit{read-only}. A cell is either read-write or read-only depending on the behavior inherited from the referenced field. A cell than has this attribute set to “\textit{true}” overrides the read-write access of the referenced field. A cell presenting a field that is read-only can not force the cell to read-write. If this attribute is not present, false is assumed.

The \textit{aready} attribute is optional. Its value is either “\textit{true}” / “\textit{1}” or “\textit{false}” / “\textit{0}”. A true value indicates that the field referenced by the cell via the \textit{idref} attribute is to acquire the \textit{aready} attribute when a new story is created. If this attribute is not present, false is assumed.

The content of a \textit{stringcell\_tag} is the text of the cell label. Each cell may have one label, the label appears in its own presentation area either to the left, right, top or bottom of the cell. The \textit{stringcell\_tag} has some attributes that affect the appearance of the label.

The \textit{length} attribute is optional. Its value is a \textit{decimal number} that, if present, defines the length in base units of the cell label. If not specified the length of the cell label is assumed to be zero.

The \textit{lstyle} attribute is optional. It defines the physical style of the cell label. If not specified the cell label inherits a style from the enclosing \textit{<row>} or \textit{<form>} tags. The \textit{lstyle} attribute value has the same values as the \textit{style} attribute.

The \textit{lalign} attribute is optional. It defines the alignment of the cell label. The value must be one of “\textit{left}”, “\textit{right}”, or “\textit{center}”. If not specified, “\textit{left}” alignment is assumed.

The \textit{lposition} attribute is optional. It defines the placement of the label relative to the cell. It must be one of “\textit{left}”, “\textit{right}”, “\textit{top}”, or “\textit{bottom}”. If not specified “\textit{top}” is assumed.

The \textit{limit} attribute is optional. Its value is either “\textit{true}” / “\textit{1}” or “\textit{false}” / “\textit{0}”. A true value indicates that the text content of the field referenced by the cell via the \textit{idref} attribute is limited to what fits into the visible space of the cell as defined by the \textit{length} attribute. If this attribute is not present, false is assumed.
The \textit{wg} attribute is optional. It defines a write group that restricts editing of the field to members of that group. It is a \textit{decimal number} which is the internal group number of the group. If not specified the cell/field has no write group assigned to it. If this attribute is present it affects the \textit{ro} attribute of the corresponding field only when that attribute is \textit{false} and the user attempting to edit the field is not a member of the group.

The \textit{mandatory} attribute is optional. Its value is either \textit{“true”} / “1” or \textit{“false”} / “0”. A true value indicates that the field referenced by the cell via the \textit{idref} attribute cannot be empty. If this attribute is not present, false is assumed.

\section{6.3 The \texttt{<checkboxcell>} tag}

\begin{verbatim}
<checkboxcell>

Function:
Defines a presentation/appearance of a checkbox field.

Attributes:
- \textit{idref} \hspace{1cm} \textit{id of a tag in the \texttt{<fields> section}}
- \textit{length} \hspace{1cm} \textit{decimal number}
- \textit{style} \hspace{1cm} \textit{”b” \ | \ “i” \ | \ “u” \ | \ “” \ | space separated list of those letters}
- \textit{align} \hspace{1cm} \textit{“left” \ | \ “right” \ | \ “center”}
- \textit{ro} \hspace{1cm} \textit{boolean_value}
- \textit{aready} \hspace{1cm} \textit{boolean_value}
- \textit{llength} \hspace{1cm} \textit{decimal number}
- \textit{lstyle} \hspace{1cm} \textit{”b” \ | \ “i” \ | \ “u” \ | \ “” \ | space separated list of those letters}
- \textit{lalign} \hspace{1cm} \textit{“left” \ | \ “right” \ | \ “center”}
- \textit{lposition} \hspace{1cm} \textit{“left” \ | \ “right” \ | \ “top” \ | \ “bottom”}
- \textit{limit} \hspace{1cm} \textit{boolean_value}
- \textit{wg} \hspace{1cm} \textit{decimal number}
- \textit{mandatory} \hspace{1cm} \textit{boolean_value}

Contains:
\textit{plain_text ?}

Used in:
\textit{row_tag}

See the description of the attributes under the \texttt{stringcell_tag} description.
\end{verbatim}
6.4 The <comboboxcell> tag

**<comboboxcell>**

**Function:**
Defines a presentation/appearance of a combobox field.

**Attributes:**
- locked [boolean value]
- idref [id of a tag in the <fields> section]
- length [decimal number]
- style "b" | “i” | “u” | "" | space separated list of those letters
- align “left” | “right” | “center”
- ro [boolean value]
- already [boolean value]
- length [decimal number]
- lstyle "b" | “i” | “u” | "" | space separated list of those letters
- lalign “left” | “right” | “center”
- lposition “left” | “right” | “top” | “bottom”
- limit [boolean value]
- wg [decimal number]
- mandatory [boolean value]

**Contains:**
- label_tag ? (entry_tag | group_tag | list_tag) *

**Used in:**
- row_tag

The `locked` attribute is optional. It determines the behavior of the combobox field. Its value is either "true" | “1” | “false” | “0”. A true value indicates only entries in the drop down list can be entered into the field. Otherwise any text can be entered into the field. If this attribute is not present, “true” is assumed.

See the description of the other attributes under the `stringcell` tag description.

6.4.1 The <label> tag

**<label>**

**Function:**
Defines the label text for a combobox field.

**Attributes:**
- None

**Contains:**
- plain_text ?

**Used in:**
- comboboxcell_tag
6.4.2 The <entry> tag

<entry>

Function:
Defines one entry of the drop down list for a combobox field.
Attributes:
- bg [decimal number]
Contains:
- plain_text ?
Used in:
- comoboboxcell_tag

The bg attribute is optional. It defines a background color that will be applied to the field when this entry is selected from the combobox list. It is a decimal number which identifies a system color. System colors are not defined in this specification. If not specified it defaults to zero and implies that no background color will be applied to the field.

6.4.3 The <group> tag

<group>

Function:
Defines a set of user names as entries for the drop down list for a combobox field.
Attributes:
- id [REQUIRED] User group name
- bg [decimal number]
Contains:
- plain_text ?
Used in:
- comoboboxcell_tag

The id attribute is required. This is the name of a user group.

The bg attribute is optional. It defines a background color that will be applied to the field when a member of the group is selected from the combobox list. It is a decimal number which identifies a system color. System colors are not defined in this specification. If not specified it defaults to zero and implies that no background color will be applied to the field.

6.4.4 The <list> tag

<list>

Function:
Defines a set of text strings as entries for the drop down list for a combobox field.
Attributes:
- id [REQUIRED] System list name
- bg [decimal number]
Contains:
- plain_text ?
Used in:
- comoboboxcell_tag

The id attribute is required. This is the name of a system list that contains zero or more text strings.

Comment: New combobox cell type.
The `bg=` attribute is optional. It defines a background color that will be applied to the field when entry in the system list is selected from the combobox list. It is a decimal number which identifies a system color. System colors are not defined in this specification. If not specified it defaults to zero and implies that the background color contained in the system list for an entry will be applied to the field.

### 6.5 The `<datecell>` tag

<table>
<thead>
<tr>
<th><code>&lt;datecell&gt;</code></th>
</tr>
</thead>
</table>

**Function:**
Defines a presentation/appearance of a date field.

**Attributes:**
- `idref`  
  *id of a tag in the `<fields>` section*
- `length`  
  *decimal number*
- `style`  
  *"b" | “i” | “u” | “” | space separated list of those letters*
- `align`  
  *“left” | “right” | “center”*
- `ro`  
  *boolean_value*
- `aready`  
  *boolean_value*
- `llength`  
  *decimal number*
- `llstyle`  
  *"b" | “i” | “u” | “” | space separated list of those letters*
- `lalign`  
  *“left” | “right” | “center”*
- `lposition`  
  *“left” | “right” | “top” | “bottom”*
- `limit`  
  *boolean_value*
- `wg`  
  *decimal number*
- `mandatory`  
  *boolean_value*

**Contains:**
- `date_value`

**Used in:**
- `row_tag`

See the description of the attributes under the `stringcell_tag` description.
6.6 The <durationcell> tag

```
<durationcell>

Function:
Defines a presentation/appearance of a time duration field.

Attributes:
- idref: id of a tag in the <fields> section
- length: decimal number
- style: "b" | "i" | "u" | "" | space separated list of those letters
- align: "left" | "right" | "center"
- ro: boolean value
- aready: boolean value
- llength: decimal number
- lstyle: "b" | "i" | "u" | "" | space separated list of those letters
- lalign: "left" | "right" | "center"
- lposition: "left" | "right" | "top" | "bottom"
- limit: boolean value
- wg: decimal number
- mandatory: boolean value

Contains:
duration_value

Used in:
row_tag
```

See the description of the attributes under the stringcell_tag description.

7 Story Fields – the <fields> tag content

7.1 The <string> tag – defining a field

```
<string>

Function:
Defines a field content.

Attributes:
- id: [REQUIRED] string
- urgency: decimal number 1 through 9
- aready: boolean value
- uec: boolean value
- ro: boolean value
- bg: decimal number

Contains:
plain_text

Used in:
fields_tag
```

The <string> and </string> tags enclose plain_text which is uniquely associated with an identifier.

The id= attribute is required. It uniquely identifies the field in the story so that it may be referenced and used. The id is referenced by the idref= attribute of a stringcell_tag or a comboboxcell_tag within the <form> element of the NSML document. Each story will have only one field with a specific id. There are some reserved ids that identify fields with specific meaning and in some cases contain system-supplied content.
<table>
<thead>
<tr>
<th><strong>Reserved field identifiers</strong></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AIR-DATE</td>
<td>decimal digits</td>
<td>Seconds since Jan 1, 1970 00:00:00 GMT</td>
</tr>
<tr>
<td>AUDIO-TIME</td>
<td>decimal digits</td>
<td>Audio read time of story in seconds. Normally based on read-rate and word-count but can be user entered.</td>
</tr>
<tr>
<td>BACK-TIME</td>
<td>special</td>
<td>Hard in-time of the story in seconds.</td>
</tr>
<tr>
<td>CA-CAPTURED</td>
<td>character string</td>
<td>Connect session information – number of characters captured.</td>
</tr>
<tr>
<td>CA-DIRECTION</td>
<td>character string</td>
<td>Connect session information – direction of connection (in or out)</td>
</tr>
<tr>
<td>CA-ELAPSED</td>
<td>character string</td>
<td>Connect session information – duration as HH:MM:SS</td>
</tr>
<tr>
<td>CA-IDENT</td>
<td>character string</td>
<td>Connect session information – session identifier</td>
</tr>
<tr>
<td>CA-ORIGIN</td>
<td>character string</td>
<td>Connect session information – originating computer name</td>
</tr>
<tr>
<td>CA-RECEIVED</td>
<td>character string</td>
<td>Connect session information – number of characters received</td>
</tr>
<tr>
<td>CA-REMOTE</td>
<td>character string</td>
<td>Connect session information – remote computer name</td>
</tr>
<tr>
<td>CA-SENT</td>
<td>character string</td>
<td>Connect session information – number of characters sent</td>
</tr>
<tr>
<td>CG-ADDR</td>
<td>character string</td>
<td>MCS / BCS – CG device address</td>
</tr>
<tr>
<td>CG-TEMPLATE</td>
<td>character string</td>
<td>MCS / BCS – CG template address</td>
</tr>
<tr>
<td>CG-TEXT</td>
<td>character string</td>
<td>MCS / BCS – CG text</td>
</tr>
<tr>
<td>CHANNEL</td>
<td>character string</td>
<td>MCS / BCS – generic event channel</td>
</tr>
<tr>
<td>CREATE-BY</td>
<td>character string</td>
<td>Name of creator of first version of the story</td>
</tr>
<tr>
<td>CREATE-DATE</td>
<td>decimal digits</td>
<td>Seconds since Jan 1, 1970 00:00:00 GMT</td>
</tr>
<tr>
<td>CUME-TIME</td>
<td>special</td>
<td>Hard out-time of the story in seconds.</td>
</tr>
<tr>
<td>DEVICE-MGR</td>
<td>character string</td>
<td>MCS / BCS – device manager name</td>
</tr>
<tr>
<td>DURATION</td>
<td>character string</td>
<td>MCS / BCS – generic event duration</td>
</tr>
<tr>
<td>EFFECT</td>
<td>character string</td>
<td>MCS / BCS – event effect</td>
</tr>
<tr>
<td>ENDORSE-BY</td>
<td>character string</td>
<td>Name of user who endorsed the story</td>
</tr>
<tr>
<td>EVENT-STATUS</td>
<td>character string</td>
<td>MCS / BCS – (video) event status</td>
</tr>
<tr>
<td>ITEM-CHANNEL</td>
<td>character string</td>
<td>MCS / BCS – playout channel for a primary event</td>
</tr>
<tr>
<td>MAIL-CC</td>
<td>character string</td>
<td>Mail information – cc address list</td>
</tr>
<tr>
<td>MAIL-TO</td>
<td>character string</td>
<td>Mail information – to address list</td>
</tr>
<tr>
<td>MODIFY-BY</td>
<td>character string</td>
<td>User name of the last modifier of the story</td>
</tr>
<tr>
<td>MODIFY-DATE</td>
<td>decimal digits</td>
<td>Seconds since Jan 1, 1970 00:00:00 GMT</td>
</tr>
<tr>
<td>MODIFY-DEV</td>
<td>character string</td>
<td>Device name on which story was last modified</td>
</tr>
<tr>
<td>MOS-ACTIVE</td>
<td>character string</td>
<td>MOS – event identifier</td>
</tr>
<tr>
<td>MOS-DURATION</td>
<td>Decimal digits</td>
<td>MOS – event duration</td>
</tr>
<tr>
<td>MOS-SUBEVENT</td>
<td>character string</td>
<td>MOS – event details</td>
</tr>
<tr>
<td>MOS-TITLE</td>
<td>character string</td>
<td>MOS -- event description</td>
</tr>
<tr>
<td>PAGE-NUMBER</td>
<td>character string</td>
<td>User entered story identifier</td>
</tr>
<tr>
<td>PRESENTER</td>
<td>character string</td>
<td>The name of the person who will read the story on-air</td>
</tr>
<tr>
<td>READY</td>
<td>character string</td>
<td>State of the story. Either READY or ?.</td>
</tr>
<tr>
<td>RESULT-INDEX</td>
<td>character string</td>
<td>Search result information – story identifier</td>
</tr>
<tr>
<td>RESULT-LOC</td>
<td>character string</td>
<td>Search result information – story location (queue name)</td>
</tr>
<tr>
<td>RUNS-TIME</td>
<td>decimal digits</td>
<td>Sum of runs times specified in production cues.</td>
</tr>
<tr>
<td>SEARCH-ID</td>
<td>character string</td>
<td>Search result information – search request identifier</td>
</tr>
<tr>
<td>STATUS</td>
<td>character string</td>
<td>Status of some element of the story.</td>
</tr>
<tr>
<td>STILL-ID</td>
<td>character string</td>
<td>MCS / BCS – still store event identifier</td>
</tr>
<tr>
<td>STILL-PRESET</td>
<td>character string</td>
<td>MCS / BCS – still store event preset</td>
</tr>
<tr>
<td>STYLE</td>
<td>character string</td>
<td>MCS / BCS – event style</td>
</tr>
<tr>
<td>TAPE-TIME</td>
<td>decimal digits</td>
<td>The run time in seconds of a tape to be played with the story</td>
</tr>
<tr>
<td>TITLE</td>
<td>character string</td>
<td>User entered story title</td>
</tr>
<tr>
<td>TOTAL-TIME</td>
<td>decimal digits</td>
<td>Total story time in seconds, sum of audio-time and tape-time</td>
</tr>
<tr>
<td>VIDEO-ID</td>
<td>character string</td>
<td>MCS / BCS – video (tape / clip) identifier</td>
</tr>
<tr>
<td>WRITER</td>
<td>character string</td>
<td>User name of the writer taking credit for the story.</td>
</tr>
</tbody>
</table>
The back-time and cume-time field content has a special encoding. The hard in/out times are expressed in seconds as either a relative time or an absolute time (a.k.a. time of day.) The time is assumed to be relative unless the first character of the field is a @ character. For example a back-time content of 600 specifies a hard in-time of 10 minutes relative to the start of the show. A back-time content of @600 specifies a hard in-time of 12:10:00 am (10 minutes passed midnight.)

The uec= attribute is optional. Its value is either “true” / “1” or “false” / “0”. A true value indicates that a user has entered content to override the system supplied content normally provided in the field. It is only significant for fields that may contain system supplied content.

The urgency= attribute is optional. Its value may be a decimal number from the 1 through 9 (if not specified “1” is assumed.) It is intended as a clue to the applications that the data in this field has some exceptional meaning and may need to be brought to the users attention. The specific method for presenting this information to the user is not defined by NSML. It could be ignored. The specific meaning of the urgency is dependent on the specific id.

The aready= attribute is optional. Its value is either “true” / “1” or “false” / “0”. A true value indicates that the content of the field affects the content of the READY field. Specifically, if any field with the aready attribute has a ? (question mark) as the first character of its content or it has no content, the READY field will have a ? as its content. If this attribute is not present, false is assumed.

The ro= attribute is optional. Its value is either “true” / “1” or “false” / “0”. A true value indicates that the field is read-only. This attribute is used in conjunction with the same attribute in the <stringcell> tag that references this <string>. Applications should only allow modifications to this <string> when the ro attributes of the <stringcell> and the <string> cell are both “false”. If this attribute is not present, false is assumed.

The bg= attribute is optional. It defines a background color that will be applied to the field. This attribute is acquired when an entry is selected from a combobox list. It is a decimal number which identifies a system color. System colors are not defined in this specification. If not specified it defaults to zero and implies that no background color will be applied to the field.

### 7.2 The <boolean> tag

| Function: | Defines a boolean field for a 0 or 1 value typically used as a checkbox field. |
| Attributes: | id [REQUIRED] string, urgency decimal number 1 through 9, aready boolean value, uec boolean value, ro boolean value, bg decimal number |
| Contains: | boolean_value ? |
| Used in: | fields_tag |

The id= attribute is required. It uniquely identifies the field in the story so that it may be referenced and used. The id is referenced by the idref= attribute of a checkboxcell_tag within the <form> element of the NSML document.
See the description of the attributes under the `string_tag` description.

### 7.3 The `<date>` tag

```
<date>

| Function: | Defines a field for a date value. |
| Attributes: | id [REQUIRED] string |
| | urgency decimal number 1 through 9 |
| | aready boolean value |
| | uec boolean value |
| | ro boolean value |
| | bg decimal number |
| Contains: | date_value ? |
| Used in: | fields_tag |
```

The `id` attribute is required. It uniquely identifies the field in the story so that it may be referenced and used. The `id` is referenced by the `idref` attribute of a `datecell_tag` within the `<form>` element of the NSML document.

See the description of the attributes under the `string_tag` description.

### 7.4 The `<duration>` tag

```
<duration>

| Function: | Defines a field for a time duration value. |
| Attributes: | id [REQUIRED] string |
| | urgency decimal number 1 through 9 |
| | aready boolean value |
| | uec boolean value |
| | ro boolean value |
| | bg decimal number |
| Contains: | duration_value ? |
| Used in: | fields_tag |
```

The `id` attribute is required. It uniquely identifies the field in the story so that it may be referenced and used. The `id` is referenced by the `idref` attribute of a `durationcell_tag` within the `<form>` element of the NSML document.

See the description of the attributes under the `string_tag` description.
8 Story Body – the <body> tag content

8.1 The <p> tag – body paragraph blocks

<p>

Function:
Defines a paragraph of text.

Attributes:
- font: string
- family: decimal number (0 – 5)
- pitch: decimal number (0 – 2)

Contains:
- content_tag *
- text_content *

Used in:
- body_tag

The <p> and </p> tags enclose the text of a paragraph. All text is inside paragraphs. See the description of the font=, family=, and pitch= attributes under the <body> tag description.

8.2 Content-based Tags

<pi> <cc>

Function:
Define the contained text as having a specific meaning.

Attributes:
None

Contains:
- text_content *

Used in:
- p_tag

Content-based tags attach a specific meaning, context or usage to the enclosed text. Applications parsing the NSML document can use these tags to do content based processing on the text.

Content-based tags may not be nested.

The <pi> and </pi> tags enclose text which are instructions to the presenter reading the story on-air. This text is not included in the timing of the story. It is not intended for display on the closed caption device and is intended for display on the prompter.

The <cc> and </cc> tags enclose text which is not read by the presenter on-air but is usually part of a package voice-over which is to be closed captioned. This text is not included in the timing of the story. It is not intended for display on the prompter.

The physical style tags may be nested in the content-based tags to create highlighting effects. Applications are free to render nested styles in content-based text in any way that matches their capabilities. If an editing application is limited in its ability to present the styles to the user in a meaningful way it is free to strip the physical styles it cannot handle.
8.3 Physical Style Text Tags

**<b> <i> <u>**

**Function:** Specify a physical style (appearance) for the contained text.

**Attributes:** None

**Contains:** text_content *

**Used in:** text_content

The physical style tags enclose text that is to be enhanced when presented to the user. Physical style tags may be nested within other physical style tags to combine highlighting effects. Applications are free to render styles in any way that matches their capabilities but the standard or preferred renderings are <b> to bold, <i> to italic and <u> to underline. If an editing application is limited in its ability to present the styles to the user in a meaningful way it is free to strip the combinations of styles it cannot handle.

These tags are unlike the content-based tags in that they do not identify the enclosed content, they are just available to the user to highlight words to improve readability.

8.4 Text Direction Tag

**<rtl>**

**Function:** Specify that the contained text should be layed out as right to left text.

**Attributes:** None

**Contains:** text_content *

**Used in:** text_content

This tag is to assist in the handling of numbers for right to left environments. This is not needed for characters that have right-to-left Unicode property. It is needed for characters that do not have that property but should be handled similarly. This is the case with digit characters.

8.5 Background color Text Tag

**<bg>**

**Function:** Specify a background color for the contained text.

**Attributes:** rgb [REQUIRED] Six hexadecimal characters.

**Contains:** text_content *

**Used in:** text_content

The <bg> tag encloses text that is to be enhanced when presented to the user. This tag may be nested within other physical style tags to combine highlighting effects. Applications are free to render them in any way that matches their capabilities.
The rgb= attribute is optional. Its value is a string of exactly six hexadecimal characters. Each pair of hexadecimal characters represents the red, green, and blue components of the color in that order.

### 8.6 Spacing and Layout Text Tags

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th>Contains</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td>pb</td>
<td>Indicates spacing and layout for text.</td>
<td>None</td>
<td>Nothing</td>
<td>text_content</td>
</tr>
<tr>
<td>tab</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>wp</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The `<pb/>` tag indicates that when printing a story a page break is to occur at that position in the text.

The `<tab/>` tag indicates that the next character is to start at the next tab stop position.

The `<wp/>` tags will appear at the positions in the text where the word wrapping was performed by the last application to modify the text. These wrap points may be used by an application if it wishes to present the text of the story with the same appearance as it last appeared.

### 8.7 The `<a/>` tag – anchor to anchored elements

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th>Contains</th>
<th>Used in</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Inserts a reference to an anchored element.</td>
<td>idref [REQUIRED] decimal number</td>
<td>Nothing</td>
<td>text_content</td>
</tr>
</tbody>
</table>

The `<a/>` tag marks the spot within a paragraph of the document body at which an anchored element is referenced. The order and position of these anchors are intended to determine the order and position of the anchored elements when displayed by an application.

The `idref` attribute is required. Its value is a decimal number that identifies a specific anchored element in the `<aeset>` that is referenced by this element. The value of the number must be between 0 and 255 inclusive.

### 8.8 The `<link>` tag

<table>
<thead>
<tr>
<th>Tag</th>
<th>Function</th>
<th>Attributes</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>link</td>
<td>Defines a hyperlink to a resource.</td>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>
8.8.1 The <show> tag

<show>

Function:
Defines a hyperlink to a resource.
Attributes:
None
Contains:
plain_text
Used in:
link_tag

The content of the <show> tag, if present, can be displayed and highlighted to denote the link.

8.8.2 The <url> tag

$url$

Function:
Defines a hyperlink to a resource.
Attributes:
None
Contains:
plain_text
Used in:
link_tag

The content of the <url> tag is the actual link to be used to retrieve or navigate to the referenced object.

9 Story Anchored Element Set – the <aeset> tag content

9.1 The <ae> tag – anchored element

<ae>

Function:
Defines an anchored element.
Attributes:
- id [REQUIRED] decimal number
- version
- type
- runs decimal number
- font string
- family decimal number (0 – 5)
- pitch decimal number (0 – 2)
Contains:

Comment: Hyperlink support.

Comment: Was hidden text.
The `<ae>` and `</ae>` tags enclose an anchored element. An anchored element is an object that is anchored at a specific position in the body of the document. How these objects are displayed is dependent on the object content and the application displaying the object. Anchored elements are referenced by anchors (`<a>` tags) within body paragraphs. The anchored elements are expected to be the same sequence as their corresponding anchors within the story body.

The `id` attribute is required. Its value is a **decimal number** that uniquely identifies the anchored element in the story so that it may be referenced and used. The `id` is referenced by the `idref` attribute of the `<a>` tag within the `<body>` element of the NSML document. Each `id` is unique within an anchored element set (`<aeset>`). The value of the number must be between 0 and 255 inclusive.

The `version` attribute is optional. Its value is **plain_text**. It is typically a News Server version number or a News Client version number of the agent that created or modified the anchored element. The iNEWS Server typically supplies `version=“S3.0”`. This can be used to account for changes between versions of the server and the client.

The `type` attribute is optional. Its value is **plain_text**. It is used to determine the type of information that may be contained in the anchored element. Known types are “G” (General), “M” (MOS), “V” (Video), “J” (Journalist Editor). If this attribute is not present “G” is assumed. Applications should allow for other types. Applications should preserve anchored elements with a `type` that is not known to the application.

The `runs` attribute is optional. Its value is a **decimal number** that represents the runs time of the anchored element. If not set it is assumed to be zero. For “G” and “M” type elements this value is the sum of all of the “RUNS=” values found in the content of the anchored element.

See the description of the `font`, `family`, and `pitch` attributes under the `<body>` tag description.

### 9.1.1 The `<hidden>` tag

**<hidden>**

**Function:**
Contains information relevant to an anchored element that is not for display.

**Attributes:**
None

**Contains:**
**plain_text**

**Used in:**
`aeset` tag

The text content is not for display. It is really metadata that is associated with the anchored element. Currently only Journalist Editor (`type="J"`) anchored elements contain such data. In any case, all applications must preserve this information when modifying a story.
9.1.2 Anchored element content – ae_content

**ae_content**

Function:
Defines the optional content elements in an ae_tag.

Start tag:
None

Attributes:
None

Contains:

- (mc_tag | ap_tag) *

Used in:

- ae_tag

9.2 The <mc> tag – machine control tag

**<mc>**

Function:
Defines a set of instructions used for machine control.

Attributes:
- error: boolean_value
- idref: id of an <attachment> tag in the <aeset-atts> section
- flags
- runs: decimal number
- status: decimal number (0 – 255)
- font
- family: decimal number (0 – 5)
- pitch: decimal number (0 – 2)

Contains:

- ap_tag *

Used in:

- ae_tag

The <mc> and </mc> tags enclose machine control instructions which are understood by Machine Control Subsystem of the News System.

The **error** attribute is optional. Its value is either “true” / “1” or “false” / “0”. A true value indicates that the News System’s Machine Control Subsystem was not able to understand or carry out the instructions contained within the tag. If this attribute is not present, false is assumed.

The **idref** attribute is optional. It identifies a specific attachment in the <aeset-atts> section which “belongs” to this machine control element. It is the id value that is matched. If not specified there is no attachment that “belongs” to this machine control element.

The **flags** attribute is optional. Its value is a sequence of characters that can be used by applications that manipulate the machine control information. Currently there can be as many as five characters. The values and meaning are:
Position | Possible Values | Meaning
--- | --- | ---
1 | Y,N | Yes/No Editor available via ActiveX
2 | Y,N | Yes/No Player available via ActiveX
3 | Y,N | Yes/No Explorer available via ActiveX
4 | A thru Z | Changes if attachment changes since last written to database. Cycles through 26 letters of English alphabet
5 | M, B | M=MOS(plain vanilla) B=CAP event (formerly BCS) (More possible values are likely to be added)

The \textit{runs=} attribute is optional. Its value is a \textit{decimal number} that represents the \textit{runs time} of the \textit{machine control element}. If not set it is assumed to be zero.

The \textit{status=} attribute is optional. Its value is a \textit{decimal number} in the range 0 to 255 that represents the status as reported from the the News System’s Machine Control Subsystem. The meanings of the values are not defined in NSML.

See the description of the \textit{font=}, \textit{family=} and \textit{pitch=} attributes under the \textit{<body>} tag description.

### 9.3 The \textit{<ap>} tag – anchored paragraphs block

\begin{table}
\begin{tabular}{|l|l|l|}
\hline
\textbf{Function:} & Defines a paragraph in anchored element. \\
\textbf{Attributes:} & \\
font & \textit{string} \\
family & \textit{decimal number (0 – 5)} \\
pitch & \textit{decimal number (0 – 2)} \\
\textbf{Contains:} & \textit{ap_content} * \\
\textbf{Used in:} & \textit{ae_tag} \\
\hline
\end{tabular}
\end{table}

The \textit{<ap>} and \textit{</ap>} tags enclose the text of a paragraph in anchored elements and machine control elements. \textit{ap_text} is differentiated from \textit{text} because \textit{ap_text} cannot contain \textit{<pb/>} or \textit{<a/>} tags.

See the description of the \textit{font=}, \textit{family=} and \textit{pitch=} attributes under the \textit{<body>} tag description.
10 Field-atts and AEset-atts Content Definition

10.1 The <attachment> tag

```
<attachment>
  Function:
  Defines a sequence of text.
  Attributes:
  id [REQUIRED] string
  Contains:
  plain_text ?
  Used in:
  field_atts_tag
  aeSet_atts_tag
</attachment>
```

The `id` attribute is required. It uniquely identifies the attachment in the story so that it may be referenced and used. It is beyond the scope of the NSML specification to define where references to the attachment reside or how they are used.

11 Example

```xml
<?xml version="1.0"?>
<nsml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://avid.com/nsml nsml.xsd"
      xmlns="http://avid.com/nsml"
      version="3">
  <head>
    <meta words="56" rate="180"/>
    <formname>RUNDOWN-AIRDATE</formname>
    <storyid>0d7bd190:00f0a7ee:43949af4</storyid>
  </head>
  <fields>
    <string id="page-number">B01</string>
    <string id="presenter">EC</string>
    <string id="title">N:UNABOMBER HOUSE</string>
    <string id="var-3">VO/W</string>
    <string id="video-id">txfer310</string>
    <string id="event-status"/>
    <string id="status">OK</string>
    <string id="item-channel"/>
    <duration id="audio-time">18</duration>
    <duration id="runs-time">22</duration>
    <duration id="total-time">40</duration>
    <date id="modify-date">1133812468</date>
    <string id="modify-by">csquire</string>
    <string id="endorse-by"/>
    <string id="cume-time"/>
  </fields>
</nsml>
```
IN OTHER NATIONAL NEWS... THE COUNTRY'S MOST FAMOUS SHACK IS ON THE MOVE TONIGHT.

THE MONTANA HOME OF UNABOMBER SUSPECT TED KACZYNSKI IS UNDER A TARP, ON THIS TRUCK, HEADING TO SACRAMENTO.

JURORS WILL BE GETTING A TOUR OF THE CABIN.

BY SHOWING HOW HE LIVED, DEFENSE ATTORNEYS ARE HOPING TO PROVE KACZYNSKI IS MENTALLY ILL.
<ap>runs 0:22</ap>

<ae id="2" version="S3.0" type="G" runs="0">
  <mc>
    <ap>CG NAT1</ap>
    <ap>NEAR GREAT FALLS, MT</ap>
  </mc>
</ae>

<ae id="3" version="S3.0" type="G" runs="0">
  <mc>
    <ap>CG NAT</ap>
  </mc>
</ae>

<ae id="4" version="S3.0" type="G" runs="0">
  <ap>wipe v/o----------------</ap>
</ae>

<attachment id="1"><![CDATA[<AttachmentContent><mos><mosID>mosid.inews</mosID><mosItemBrowserProgID><mosAbstract>Unabomber (VO)</mosAbstract><objID>60A10000</objID><objSlug>Unabomber</objSlug><objTB>60</objTB><objDur>1323</objDur><itemEdDur>1</itemEdDur><itemID>1</itemID></mos></AttachmentContent>]]></attachment>