



CSS3 – La gestione dei testi

DARIO DE JUDICIBUS

Un modulo particolarmente intrigante per la gestione di scritti non occidentali che fornisce tuttavia proprietà interessanti per ottenere effetti speciali anche nella nostra lingua. Tutto questo grazie allo standard Unicode.

Uno dei moduli CSS3 che presenta il maggior numero di novità è quello relativo al testo. Il testo, infatti, rappresenta in genere il contenuto principale di un documento, a meno che questi non sia esplicitamente orientato alla multimedialità. La maggior parte delle pagine web è infatti formata prevalentemente da testi, anche se la grafica riveste comunque una componente importante nella pubblicazione di un sito in Internet. Fino ad oggi, tuttavia, il linguaggio CSS dedicava ai testi poche proprietà, tendenzialmente di carattere decorativo. A parte quindi i vari `text-decoration`, `text-shadow` e `text-transform`, le sole altre proprietà inerenti al testo riguardavano l'allineamento (`text-align` e `text-indent`) e la spaziatura (`letter-spacing`, `word-spacing` e `white-space`), queste ultime peraltro poco utilizzate. È importante qui non confondere le proprietà del testo con quelle dei tipi di caratteri: proprietà come `font-weight` o `font-size` riguardano un altro modulo, quello appunto dei tipi (*font*); quando parliamo di proprietà del testo, invece, si intende il modo in cui viene rappresentato uno scritto, *indipendentemente* dal tipo di carattere scelto.

Ad ogni modo, con la versione 3 del CSS si è voluta dare maggiore enfasi alla rappresentazione di testi resi con alfabeti e regole di scrittura differenti da quella occidentale, quest'ultima denominata più semplicemente *latina*, anche se le sue estensioni comprendono molti altri caratteri come quelli nordici o di molti Paesi dell'Est, quali ad esempio `å`, `þ`, `ß`, `ö` e `ğ`.

Ma qual è l'utilità per noi?

A questo punto è opportuno aprire una parentesi. Per quale motivo a noi occidentali dovrebbero interessare le regole relative a una perfetta resa dell'arabo o del cinese in una pagina web? I motivi sono essenzialmente due. Il primo è che oggi sempre più aziende iniziano ad avere se non l'intero sito, almeno una o due pagine tradotte in altre lingue. A parte l'inglese, lo spagnolo e altre lingue europee, oggi sempre più importanti per fare affari con l'estero, si è iniziato già da alcuni anni a produrre pagine in russo e oggi ci si

sta focalizzando sul cinese, dato che la Cina sta diventando sempre di più un mercato interessante per le industrie occidentali. Ad esclusione tuttavia delle grandi aziende, che si possono permettere di far sviluppare un sito direttamente sul posto, le piccole e medie imprese affrontano in genere questo problema chiedendo a chi ha sviluppato il loro sito in Italia di pubblicare anche un paio di pagine in altre lingue. E qui viene fuori il problema: se trovare un buon traduttore professionista è tutt'altro che difficile, non si può pretendere che questi sappia anche scrivere il codice XHTML e i fogli di stili necessari per rendere correttamente quanto tradotto. Nasce così l'esigenza per lo sviluppatore di conoscere almeno le fondamenta di come si debbano presentare testi nelle lingue del Medio ed Estremo Oriente, in modo da rappresentare la traduzione nel modo corretto.

C'è poi un secondo motivo. Alcune delle proprietà utilizzate per le lingue orientali danno un maggiore controllo del testo rispetto a quello fornito dalle versioni precedenti del linguaggio CSS anche per la scrittura latina, tanto da avvicinarsi notevolmente a linguaggi molto avanzati come il TeX. Questo permette di sviluppare documenti particolari e di realizzare tecniche di composizioni altrimenti possibili, appunto, solo con linguaggi di quel tipo. Vedremo in questo articolo alcune applicazioni a documenti occidentali delle proprietà avanzate per i testi orientali.

Chiusa la parentesi.

I principi della scrittura

Prima di entrare in merito alle singole proprietà, è opportuno fare un breve cenno di quali siano le regole fondamentali di una *scrittura*, ovvero dello scrivere usando un determinato alfabeto e specifiche regole ortografiche. Ogni lingua, infatti, ha le sue regole che non si esauriscono semplicemente con l'utilizzo di un alfabeto differente dal nostro ma riguardano vari aspetti, come la composizione delle parole, il verso di scrittura, la progressione del testo, la punteggiatura, la spaziatura e via dicendo.

D 0044	T 0054	d 0064	t 0074
E 0045	U 0055	e 0065	u 0075
F 0046	V 0056	f 0066	v 0076

Caratteri base dell'alfabeto latino

Ä 00C4	Ô 00D4	ä 00E4	ô 00F4
Å 00C5	Õ 00D5	å 00E5	õ 00F5
Æ 00C6	Ö 00D6	æ 00E6	ö 00F6

Caratteri supplementari dell'alfabeto latino

Consideriamo innanzi tutto la progressione del testo, ovvero in che verso viene redatto, e conseguentemente letto, un testo. Ogni testo può essere visto come una successione di blocchi al cui interno si sviluppa una successione di parole. Le parole, a loro volta, sono formate da segni, detti in linguaggio tecnico *glifi*, i quali si possono anche combinare insieme a seconda della loro posizione nella parola. Un caso classico nella scrittura latina sono le *legature*.

f + i = fi

legatura latina: effe più i

ح + ر = حر

legatura araba: hah più reh

Da notare che un glifo è qualcosa di differente da un carattere. Lo stesso carattere può essere reso da vari glifi, ovvero da vari segni, a seconda del contesto. Ad esempio, molti caratteri arabi sono resi da glifi differenti a seconda che si trovino all'inizio, alla fine, in mezzo alla parola o piuttosto isolati.

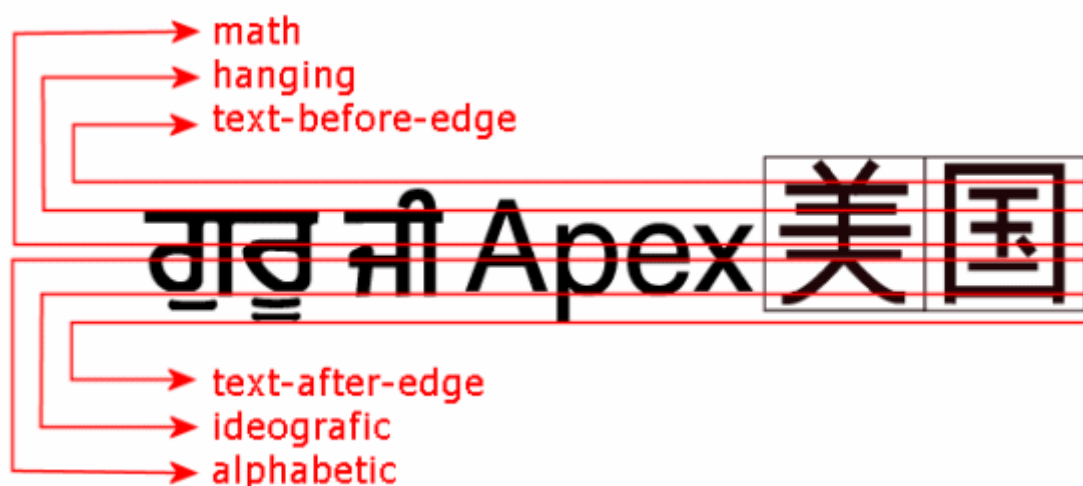


Altre volte è un singolo glifo a rappresentare più caratteri allo stesso tempo. Ad esempio i caratteri **f** e **i** possono essere resi separatamente (due glifi) oppure come un unico glifo nella legatura **fi**. Analogamente, il carattere arabo *hah* seguito dal carattere *reh* genera un glifo differente dalla semplice sovrapposizione dei due.

In molte lingue, poi, alcuni caratteri vengono rappresentati come segni diacritici al di sopra o al di sotto di altri, come succede per gli accenti nella nostra lingua. Un esempio sono le vocali in ebraico e in arabo.



Questo, ovviamente, porta a una ridefinizione ai concetti di linea base (*baseline*) e di spaziatura in scritture differenti da quella latina.



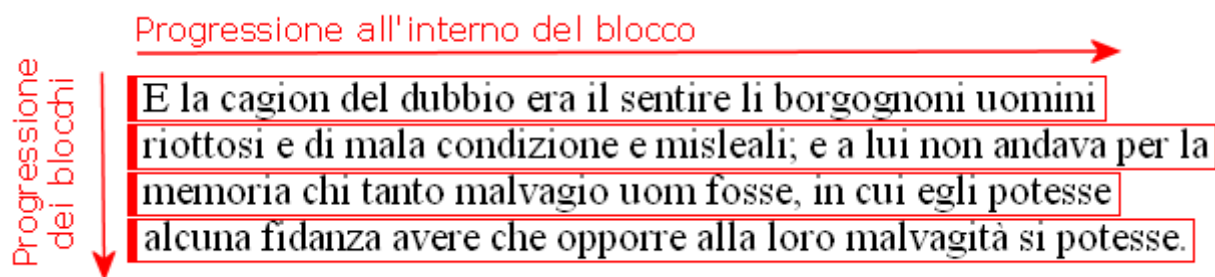
Quando parliamo di progressione del testo, quindi, dobbiamo indicare come il testo si sviluppa all'interno di un blocco e come si posizionino i blocchi all'interno della pagina. Ad esempio, in italiano il testo viene scritto sotto forma di righe orizzontali (blocchi) che si susseguono dall'alto in basso; poi, all'interno di ogni riga, il testo procede da sinistra a destra. In questo caso si dice che il *flusso* è orizzontale. Similmente, in arabo, il testo procede da destra a sinistra, sempre in blocchi orizzontali che scendono dall'alto verso il basso. In giapponese, invece, i blocchi possono essere verticali e procedere da destra a sinistra, con il testo che si sviluppa dall'alto verso il basso. In quest'ultimo caso diremo che il *flusso* è verticale.

La progressione del testo

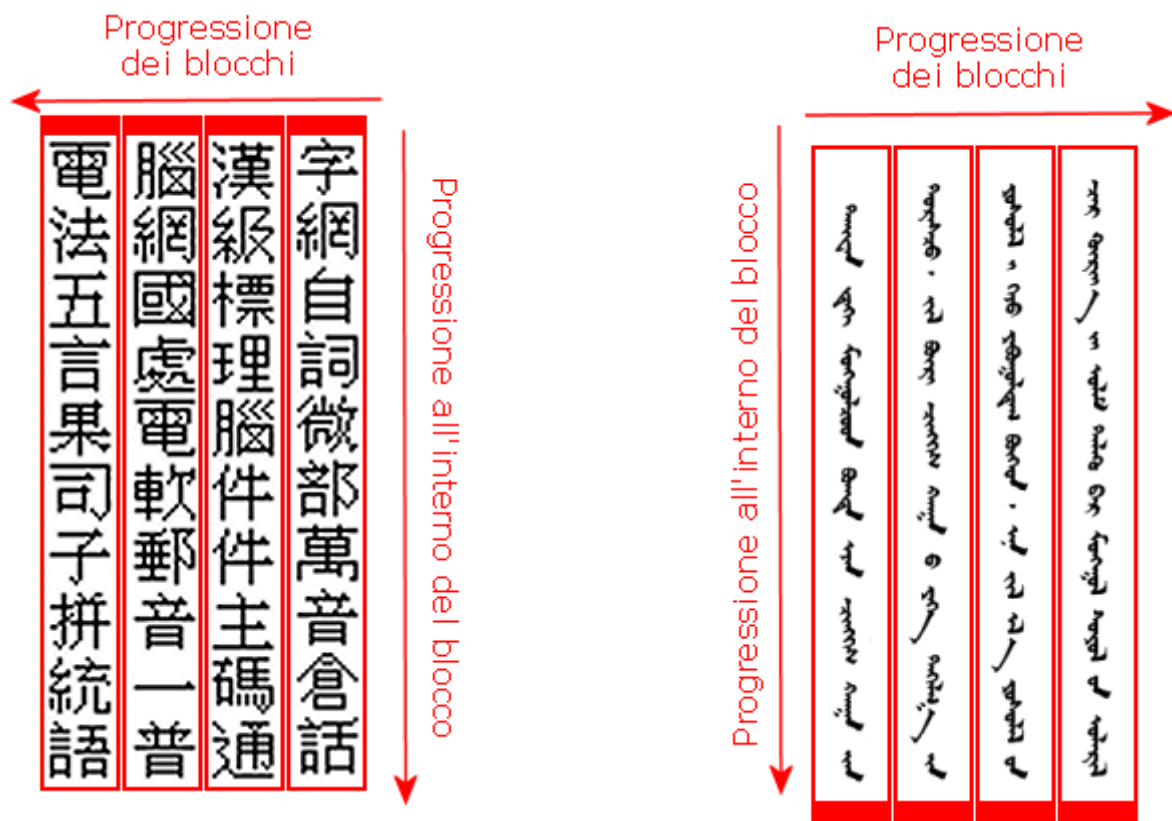
In CSS3, la proprietà **block-progression**, che specifica la successione dei blocchi, può quindi assumere tre valori: **tb**, ovvero dall'alto in basso per i flussi orizzontali, **lr** ed **rl**, ovvero da sinistra a destra e da destra a sinistra rispettivamente per quelli verticali. Viceversa **direction**, che riguarda invece il verso del testo nel blocco, può assumere solo due valori: **ltr** ed **rtl**. Il significato di questi valori va inteso relativamente all'orientamento

del flusso. Ad esempio, per un flusso verticale come **rl**, **ltr** indica che il testo si estende nel blocco dall'alto verso il basso. Questo è appunto il tipico caso del giapponese.

In linea di massima, la coppia di valori (**ltr**, **tb**) è tipica di tutti i testi occidentali, inclusi quelli in cirillico e greco, (**rtl**, **tb**) caratterizza i testi mediorientali come l'arabo, il farsi e l'ebraico, (**ltr**, **lr**) quelli dell'estremo oriente, come il giapponese e il cinese e infine (**rtl**, **lr**) il mongolo. Da notare che sia il cinese che il mongolo hanno un flusso verticale dall'alto in basso, ma dato che i blocchi per il primo scorrono da destra a sinistra e per il secondo da sinistra a destra, nel primo caso la sinistra del blocco è in alto, nel secondo in basso.



Italiano: testo da sinistra a destra in blocchi dall'alto in basso

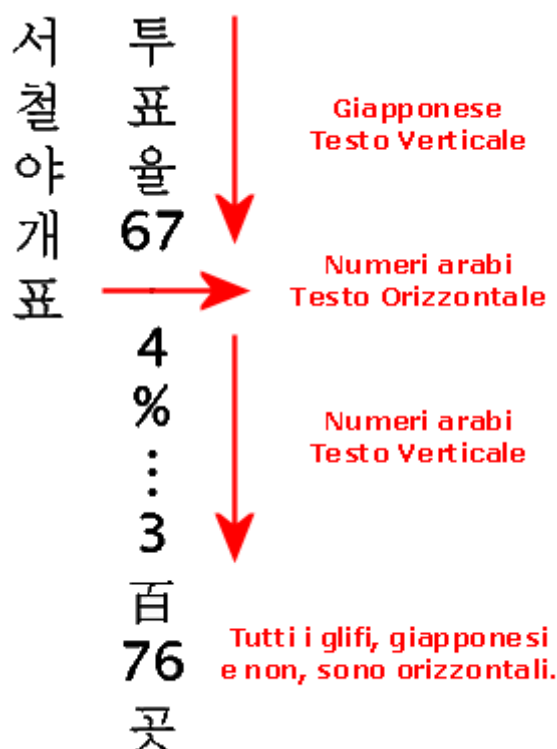


Cinese: testo da sinistra a destra (del blocco) in blocchi da destra a sinistra

Mongolo: testo da destra a sinistra (del blocco) in blocchi da sinistra a destra

C'è poi da considerare l'orientamento del singolo glifo rispetto al modo normale di disegnarlo. Per specificare l'orientamento di un glifo si usano **glyph-orientation-vertical** e **glyph-orientation-horizontal**, a seconda che il flusso sia verticale od orizzontale. Questo perché in un testo giapponese, ad esempio, può capitare che un

termine occidentale possa essere reso sia verticalmente che orizzontalmente. Quest'ultimo caso è chiamato *tate-chu-yoko*. Sono possibili diversi valori, quattro fissi, ovvero nei quali si specifica l'angolo che forma il lato inferiore del glifo rispetto al lato inferiore del blocco (**0deg**, **90deg**, **180deg** e **270deg**) e tre predefiniti (**auto**, **upright** e **inline**) che danno risultati differenti a seconda del valore del tipo di progressione.

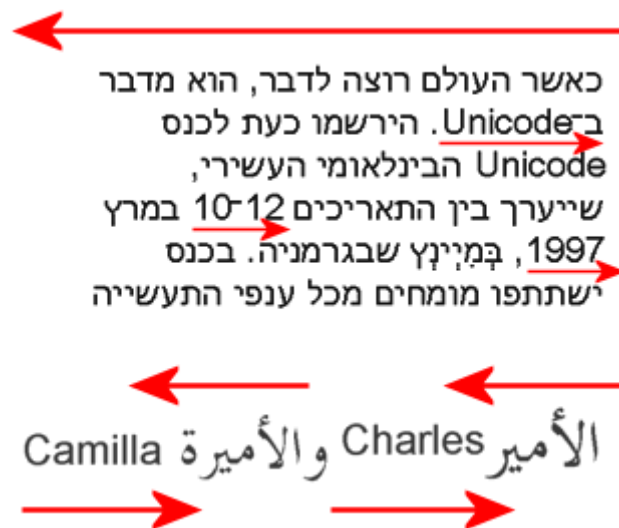


Ovviamente questi meccanismi possono essere utilizzati anche per visualizzare testi in scrittura latina. Ad esempio per generare titoli verticali con le lettere una sopra l'altra in verticale, come nell'insegna di un albergo, oppure una accanto all'altra, orizzontalmente, ovvero semplicemente ruotando l'intera riga di testo di 90°.

RUOTATO DI 270°
VERTICALE
VERTICALE
RUOTATO DI 90°

Sempre in relazione al flusso del testo, il CSS3 mette a disposizione un'altra proprietà chiamata **unicode-bidi**, che permette di controllare il tipo di algoritmo da utilizzare quando in uno stesso testo sono presenti parole che utilizzano modalità di scrittura

differenti, come ad esempio l'italiano e l'arabo, tenendo presente che è alquanto differente considerare un inciso in arabo in un testo italiano piuttosto che un inciso in italiano in un testo arabo. In genere l'UBA (*Unicode Bidirectional Algorithm*) sa da solo come gestire queste situazioni, ma quando il testo è molto complesso o contiene più incisi in varie lingue, può essere necessario forzarne il funzionamento aprendo un ulteriore livello di *annidamento*.



Infine c'è la proprietà **text-script** che serve ad indicare esplicitamente quale tipo di scrittura si stia utilizzando, dato che questa non incide solo sulla progressione del testo, ma anche su altri aspetti, come la spaziatura dei caratteri e la gestione della punteggiatura. In genere il *browser* dovrebbe riconoscere automaticamente qual è l'algoritmo giusto da applicare a seconda dell'alfabeto utilizzato, ma non sempre è così, e comunque ad alcuni linguaggi, come al cinese o al giapponese, possono essere applicate regole differenti di scrittura.

Il posizionamento del testo

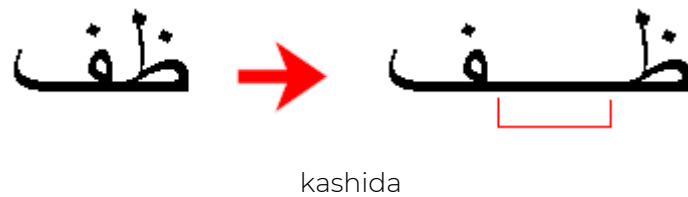
Anche per quanto riguarda l'allineamento del testo, ovviamente, la possibilità di trattare modalità di scrittura differenti ha comportato una serie di modifiche ai valori che potevano assumere alcune proprietà presenti anche nelle versioni precedenti del CSS.

Tanto per cominciare, in aggiunta alla prospettiva *destra/sinistra* se n'è aggiunta un'altra che vede ragionare sul testo in termini di *inizio/fine* e *prima/dopo*. Se il *prima* sia a destra o a sinistra, ovviamente, dipende a questo punto dalla modalità di scrittura scelta.

Per quanto riguarda l'allineamento (**text-align**), quindi, oltre a **left**, **right**, **center** e **justify**, abbiamo ora **start** ed **end**. Quest'ultimi valori indicano che il testo debba essere allineato rispettivamente all'inizio o alla fine della progressione all'interno del singolo blocco. Anche **left** e **right** comunque vanno reinterpretati a seconda della modalità di scrittura. Se infatti il testo è orizzontale, il loro significato rimane immutato, ma se è verticale, la destra e la sinistra dipendono dalla progressione dei blocchi verticali. Quelli da sinistra a destra hanno infatti il lato sinistro del blocco in basso, quelli da destra a sinistra in alto. Per le celle di una tabella, inoltre, è anche possibile impostare l'allineamento a una

qualsiasi stringa di testo, possibilità usata soprattutto nell'allineamento al separatore dei decimali.

In aggiunta all'allineamento, ora esiste anche un'altra proprietà che riguarda specificatamente la giustificazione del testo: **text-justify**. Le varie forme di scrittura, infatti, hanno modalità differenti di gestire un'eventuale spaziatura all'interno del blocco per garantire l'allineamento sia all'inizio che alla fine dello stesso. Non entreremo qui nei particolari: ci limiteremo a dire che in alcune lingue esiste il concetto di parola, per cui si può aumentare la spaziatura fra le parole; in altre la spaziatura può essere aumentata anche fra i singoli glifi, sia che siano ideogrammi, sia che siano simboli fonetici; in altre ancora, come l'arabo, si può "stirare" la parola allungandola in modo da permettere la giustificazione (*kashida*).



Alcune di queste tecniche di spaziatura, come ad esempio **distribute**, possono essere utilizzate anche con la scrittura latina per ottenere particolari effetti specialmente sulle intestazioni della pagina. Da notare l'aggiunta nel CSS3 di una nuova proprietà, **text-align-last**, che stabilisce come vada allineata l'ultima riga di un testo.

Senza entrare ulteriormente nei dettagli, fra le altre proprietà relative al posizionamento del testo segnaliamo il qualificatore **hanging** che permette di indentare non solo il primo, ma tutti i successivi blocchi di testo della quantità specificata da **text-indent**. Si tratta di una possibilità molto utile anche nella scrittura latina, dove finora l'effetto *blockquote* veniva realizzato con il margine (*margin*) o il sovrappiù (*padding*).

Andare a capo

Un altro problema non di second'ordine è come si vada a capo a fine riga. In italiano la regola più semplice è andare a capo a fine parola o all'occorrenza di un eventuale trattino. Se poi si conosce la sillabazione fra le parole, allora si possono spezzare anche le singole parole. In altre forme di scrittura, tuttavia, le cose sono più complesse. Ad esempio, in cinese un singolo ideogramma può rappresentare una parola e tuttavia ci sono ideogrammi che non possono essere separati su due righe. In thailandese, poi, non esistono proprio gli spazi fra le parole, per cui, o si introduce fra una parola e l'altra uno spazio di larghezza nulla, in modo che il browser abbia un punto di riferimento per la suddivisione in righe, o si ha a disposizione un vocabolario per sapere come spezzare il testo su più blocchi. C'è infine il problema delle parole molto lunghe che, in assenza di un dizionario, è difficile capire come vadano spezzate.

Il CSS3 mette a disposizione varie proprietà per controllare tutti questi aspetti, come **word-break-inside**, che permette o meno di spezzare una parola. Ovviamente, nel caso questa

proprietà assuma il valore **hyphenate**, il browser dovrà essere dotato di un apposito dizionario relativo alla lingua utilizzata nel testo.

Sul fatto o poi se il testo debba o meno andare a capo, la questione è definita attraverso un altro gruppo di proprietà, prima fra tutte **wrap-option**. Questa può assumere i valori **wrap**, **no-wrap**, **soft-wrap** ed **emergency**. Tutto il discorso si basa sul concetto di *preserved line-feed*, ovvero di “ritorno a capo garantito”. In pratica, in un documento HTML, nel testo sorgente possono essere presenti uno o più caratteri di ritorno a capo. In genere questi sono il carattere U+000A oppure la coppia U+000D U+000A. Tuttavia è anche possibile che il testo contenga il separatore di linea Unicode U+2028 e il separatore di paragrafo U+2029. Quale che siano i caratteri utilizzati, quando il browser incontra un carattere che rimanda a capo nel codice sorgente deve decidere cosa fare. A tal scopo viene utilizzata la proprietà **linefeed-treatment**. Se questa assume il valore **ignore**, i caratteri di ritorno a capo sono ignorati e rimossi, mentre vengono appunto “garantiti” se il valore è **preserve**; nel caso il valore sia **treat-as-space** o **treat-as-zero-width-space** vengono sostituiti da spazi normali (U+0020) o a larghezza nulla (U+200B); infine se il valore è **ignore-if-after-linefeed**, ogni ritorno a capo che segue immediatamente un altro ritorno a capo viene ignorato. Il valore **auto** agisce come **ignore**, **treat-as-space** o **treat-as-zero-width-space** a seconda della lingua nella quale è scritto il testo. A questo punto, se **wrap-option** assume il valore **wrap**, il testo va a capo sia a fronte di ogni ritorno a capo garantito, sia quando più opportuno in funzione dei valori di line-break e word-break, mentre se assume il valore **no-wrap** il testo va a capo solo a fronte di ritorni “garantiti”. Altri valori servono a gestire situazioni più complesse. Nel caso poi il testo non vada a capo, ci pensa la proprietà **text-overflow** a specificare come vada resa la “troncatura” a fine blocco: **clip** si limita a non visualizzare il testo eccedente, **ellipsis** fa terminare il testo con un qualche simbolo che indica la troncatura, come ad esempio i tre puntini, mentre **ellipsis-word** fa lo stesso ma senza mai spezzare una parola, ovvero il simbolo di troncatura è posto dopo l’ultima parola che entra nel blocco.

Lorem ipsum dolor sit amet, consectetur...

Tutte queste proprietà sono ovviamente molto utili anche nella scrittura latina. Ad esempio, in un *blog* si può creare un riquadro che contiene la lista degli ultimi commenti pubblicati in modo che ogni commento stia su una singola riga e che se il testo è troppo lungo, questo venga troncato e fatto seguire dal carattere *tre puntini* (...).

La spaziatura

Per quanto riguarda la spaziatura, sia **letter-spacing** che **word-spacing** sono rimaste sostanzialmente le stesse del CSS2. Nella versione 3 sono state tuttavia aggiunte le proprietà **punctuation-trim**, **text-kerning** e **text-autospace** che tengono conto delle necessità tipiche dei linguaggi orientali, soprattutto quelli ideografici. In particolare la seconda tiene conto dell’esigenza di separare maggiormente eventuali cifre nella

numerazione araba, come le date, ad esempio, dagli ideogrammi che le precedono e le seguono.

La decorazione

Come abbiamo detto all'inizio dell'articolo, gli effetti decorativi sono stati quelli che inizialmente hanno giustificato l'esistenza di questo modulo, almeno nel CSS1 e CSS2. Sebbene nel CSS3 si sia data maggior enfasi alla gestione vera e propria del testo, soprattutto in relazione a tipi di scrittura non occidentali, anche queste proprietà hanno subito dei miglioramenti nella terza versione di questo linguaggio. In particolare, con il CSS3 è ora possibile scegliere molti più tipi di sottolineature e di controllare come la linea si deve comportare a fronte delle caratteristiche del testo.

solid
double
dotted
dashed
dot dash
dot dot dash
wave

Sebbene anche in questo caso alcune proprietà sono state pensate essenzialmente per gli alfabeti orientali, valori come **after-edge** per la proprietà **text-underline-position** possono essere utilizzati anche con l'alfabeto latino, ad esempio per evitare che la linea tagli le gambette di alcuni caratteri come **p** o **g**.

Griglie

In molti alfabeti orientali, come il Cinese o il Giapponese, i caratteri sono scritti all'interno di una griglia immaginaria. Questo concetto non è nuovo neanche a noi occidentali, dato che quando scriviamo un programma o una tabella spesso usiamo un carattere a spaziatura non proporzionale, in modo che i vari caratteri siano allineati anche verticalmente, indipendentemente dallo spazio che occupano in larghezza.

本	次	会	议	将	汇	集	各	方	面	的	专	家	。
涉	及	的	领	域	包	括	:	国	际	互	联	网	和
统	一	码	,	国	际	化	和	本	地	化	,	统	一
码	在	操	作	系	统	和	应	用	软	件	中	的	实
现	,	字	型	,	文	本	格	式	以	及	多	文	种
计	算	等	。										

Nel CSS3 sono state previste alcune proprietà proprio per gestire questa modalità di scrittura: **line-grid**, **line-grid-mode** e **line-grid-progression**.

Combinazioni

Per ultima segnaliamo la proprietà `text-combine` che permette di combinare più caratteri nello spazio di uno solo, cosa molto comune nei linguaggi orientali. Ad esempio, in giapponese i caratteri possono essere raggruppati in singoli blocchi (*kumimoji*) o in linee (*warichu*).

ボルダー市 (アメリカ合衆国
コロラド州の町) の人口は 95,000 人です。
└──────────┘
warichu

Questa proprietà può essere usata anche con l'alfabeto occidentale per ottenere particolari effetti, come ad esempio eventuali annotazioni in linea.

L'uomo giunse al DCAS (Dip. Complicazioni
Affari Semplici)
e lì chiese di parlare al Direttore.