Propriété / Attribut

Ca m'épate qu'on puisse vouloir se passer volontairement des propriétés automatiques... Si on commence à dire qu'on veut garder des champs privés "parce que ça peut servir", alors autant remettre le constructeur par défaut "parce que ça peut servir", autant implémenter IDIsposable "parce que ça peut servir", autant définir explicitement add/remove sur les event pour la même raison http://www.developpez.net/forums/images/smilies/icon_smile.gif  
  
Donc, pourquoi les propriétés plutôt que les champs ?

* + Parce qu'on ne peut pas définir de champ dans une interface
  + parce qu'on ne peut pas rendre un champ virtuel
  + parce que plein d'API faisant appel à la réflexion (sérialisation, binding, ...) ne gère que les properies et pas les champs
  + parce que c'est un breaking change, comme l'a dit DonQuiche, et donc une lib B qui référence A ne marchera plus sans être recompilée si un champ de A devient une propriété
  + parce que d'un point de vue perf, on s'en fout, c'est le jitter qui se chargera d'inlier tout ça
  + parce que ça montre bien qu'une classe, en tant qu'unité d'encapsulation, ne donne jamais accès à son état interne sans se laisser la possibilité de valider/notifier/whatever
  + Parce que ça normalise ce qu'expose une classe/une interface: des propriétés, des méthodes, des events, et c'est tout.
  + Parce que c'est le standard, et que si chacun fait à sa sauce pour ce genre de choix consensuel, ça engendre un surcoût projet inutile en temps d'adaptation, refactoring, ...
  + lever des évèvènement au changement de valeur d'une propriété etc..., mais pas directement avec un champs.

Pourquoi les propriétés automatiques plutôt que les propriétés manuelles (dans le cas of course où on fait rien dans le get set)

* + Moins verbeux!
  + pas de problématique de convention de nommage (je mets un underscore? une minuscule? du hongrois?
  + parce que l'intellisense n'est pas pollué à l'intérieur de la classe par deux fois plus de tokens que nécessaire