TABELL 1. Kortfattad beskrivning av de flesta variablerna i request_id-objektet

Variabel	*Förklaring*

list (string) supports;	Vad klienten kan hantera.
	Innehållet i den här listan kommer från filen etc/supports, per default. Det hela kan ändras i konfigurationsinterfacet.
	Används lämpligtvis för att generera olika kod till olika klienter
	Exempel:
	`if(request_id->supports->tables)`
	`  return make_table();`
	`else`
	`  return make_pre();`

string remoteaddr;	IP-nummret till datorn på andra sidan förbindelsen, som en sträng (`"158.126.90.157"`)
array (string) client;	Vilken klient som frågar efter sidan, en array eftersom protokollet (HTTP/1.*) dumt nog tillåter flera User-Agent: header rader, och man vet ju aldrig.. Lämpligtvis använder man '`*`'-operatorn för att få en sträng av det hela (`request_id->client*""`). Ofta är det dock mycket smartare att använda supports istället för client.
array (string) referer;	Den (eller de..) sidor som refererade till den här sidan.
list (string) pragma;	Alla pragmaheadrar klienten har skickat. Intressant är '`no-cache`', som skickas när man trycker på 'reload' av de flesta klienter.
string prot;	Det protokoll som användes för att generera frågan, troligtvis HTTP/1.0 eller HTTP/1.1, men även saker som FTP, GOPHER och HTTP/0.9 kan förekomma.
string method;	Metod, troligtvis `"GET"`.
string rest_query;	Allt efter ett `?` i URL-en som inte är variabler.
string raw;	Hela requesten som rå data.
string query;	Allt efter `?` i URLen.
string not_query;	Allt innan `?` i URLen exklusive prestate, det är det här som används av spinner för att få fram vilka moduler som requesten ska mappas genom.
string data;	Allt i en BODY i requesten. Oftast inte mycket, men vid metoden POST används det till alla variabler.
array (int|string) auth;	Antingen • 0 (ingen authentifiering skickad av klienten) • `({ 0, "`*`username:password`*`" })` (authentifiering skickad, men auth modulen tycker inte att den är korrekt, och användaren finns inte ens) • `({ 0, "username" })` (authentifiering skickad, men auth modulen tycker inte att den är korrekt, och användaren finns) • `({ 1, "username" })` (authentifiering skickad, och auth modulen tycker att den är korrekt)
mapping (string:string) cookies;	Alla cookies som klienten har skickat.

TABELL 1. Kortfattad beskrivning av de flesta variablerna i request_id-objektet

Variabel	*Förklaring*
int time;	När gjordes uppkopplingen? Sekunder sedan `00:00:00 den 1/1 1970.`
object conf;	Du behöver nog inte den här variabeln, om du inte vill utföra udda operationer.
string raw_url;	URLen helt oparsad som den kom från klienten.
mapping (string:string) variab-les;	I HTTP finns det en kul sak som heter 'variabler', de kommer oftast från en 'form'. I den här mappingen finns allihop instoppade, färdigparsade. Exempel. En förfrågan efter filen `/goo/bar?hmm=hej&foo=%20` kommer att resul-tera att variables sätts till `([ "hmm":"hej", "foo":" "]).`
mapping (string:array (string)) misc;	Blandat smått och gott. Du behöver nog inte bry dig, men man kan ju alltid skriva ut fältet för att se vad det innehåller: `return sprintf("%O", request_id->misc);` Här kan du även definiera dina egna variabler, om du mot all förmodan behöver det. Du bör dock se till att dina variabler har unika namn, om du inte vill råka ut för otrevliga överraskningar. Variablerna är request-lokala.

list (string) prestate;	Prestate används mycket, iallafall på Lysator.
	Det är en lista med strängar, man kan alltså göra saker som
	`if(request_id->prestate->nobg)`
	`  no_background = 1;`
	Prestate skrivs innan filnamet I URLen, så här: /(foo,bar)/goo/bar

list (string) config;	Config fungerar precis som prestate, men det lagras inte undan i URLen, utan i client-side cookies. Detta har flera fördelar: • Den syns inte i URL-en • Användaren kommer att ha kvar sin config även om hon avslutar sin Netscape och startar den igen. • Om folk lägger upp länkar till dina sidor i sina sidor kommer de inte att inklu-dera konfigurationen. Men även några nackdelar: • Det kan bara finnas en lista med konfigurerade val per server. • Det är svårt att bli av med sin konfiguration. • Bara Netscape implementerar dem än. För att addera något till 'config' låter man användaren accessa en fil enligt: `http://din.server:port/<+config,-config,...>/riktig/URL.` Användaren kommer då att få configen tillagd till cookien Spinner-Config i sin klient, och få en redirect till /riktig/URL. Detta är mycket Spinner specifikt. Man kan också använda SPML för att göra det `<aconf +/-CONFIG></aconf>`.