qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm

|  |
| --- |
| Υλοποίηση RTree  εργασία στα πλαίσια του μαθήματος ανάκτησης δεδομένων  2/15/2008  Κρητικός Απόστολος (914) & Σκαλιστής Στέφανος (1024) |

# Το R-δένδρο

Αν και γνωστό από ήδη από την θεωρία θα γίνει συνοπτική αναφορά. Το R-δένδρο είναι μία δενδρική δομή για πολυδιάστατα δεδομένα. Στη συγκεκριμένη υλοποίηση το δένδρο δεν αποθηκεύει ορθογώνια, όπως ορίζεται στο άρθρο Guttman, αλλά σημεία. Το δένδρο υποστηρίζει τρείς βασικές λειτουργίες:

* Εισαγωγή σημείου στο δένδρο.
* Ερώτημα διαστήματος με βάση ένα σημείο αναφοράς p στο n-διάστατο χώρο και μία απόσταση d, όπου πρέπει να βρεθούν όλα τα σημεία, τα οποία περιέχονται στη δομή, που απέχουν το πολύ d από το σημείο d.
* Ερώτημα k-Κοντινότερων Γειτόνων με βάση ένα σημείο αναφοράς p και έναν ακέραιο αριθμό k, όπου πρέπει να ανακτηθούν τα k-Κοντινότερα σημεία στο p.

(Σημείωση: Συμφώνα με τη γενικότερη θεωρία των k-Κοντινότερων Γειτόνων αν υπάρχει ισοπαλία στις τελευταίες θέσεις πρέπει να επιστραφούν όλα τα ισόπαλα σημεία).

Τέλος πρέπει να κρατάει στοιχεία όπως το ύψος του δένδρου και το πόσα σημεία είναι αποθηκευμένα στη δομή αυτή.

# Δομές και τεχνικές λεπτομέρειες της υλοποίησης

Όλοι η δομή και η λειτουργία του δένδρου περιγράφεται αναλυτικά στο JavaDoc το οποίο παρατίθεται. Ακολουθεί μια συνοπτική περιγραφή των τεχνικών λεπτομερειών. Για περαιτέρω λεπτομέρειες ανατρέξτε στην αναλυτική περιγραφή του JavaDoc.

Οι κόμβοι:

Το δένδρο αποτελείται από εσωτερικούς κόμβους και κόμβους-φύλλα. Οι εσωτερικοί κόμβοι είναι οι τυπικοί, έχοντας μόνο ένα πίνακα με τις καταχωρήσεις (entries) του και το ελάχιστο περικλείον ορθογώνιο (minimum bounding rectangle) που περιέχει τις καταχωρήσεις αυτές. Τα φύλλα από τη άλλη πλευρά, έχουν επιπρόσθετα έναν αριθμό σελίδας, ο οποίος υποδηλώνει σε ποια σελίδα κείτονται τα σημεία τα οποία του έχουν ανατεθεί. Πρέπει να σημειωθεί ότι για οποιαδήποτε ενέργεια στα φύλλα, πρέπει πρωτίστως να φορτωθεί η σελίδα (ευθύνη του Διαχειριστή Δίσκου), να εκτελεστούν οι ενέργειας και έπειτα να αποδοθεί η σελίδα στο σύστημα.

Η εισαγωγή σημείου:

Κατά την εισαγωγή, χρησιμοποιούνται δύο στοίβες με σκοπό την ανοδική (ή προς τα πίσω) διάσχιση. Στη μία εξ αυτών αποθηκεύεται ο κόμβος-γονιός ενώ στην άλλη αποθηκεύεται ένας ακέραιος ο οποίος υποδηλώνει την θέση του πίνακα καταχωρήσεων στον κόμβο-γονίο η οποία χρησιμοποιήθηκε για να ανακτηθεί ο κόμβος-παιδί.

Οι διασπάσεις κόμβων και η αναπροσαρμογή του δένδρου εκτελούνται ακριβώς όπως περιγράφονται στο άρθρο του Guttman και δεν θα σχολιαστούν περαιτέρω.

Ερώτημα Περιοχής:

Στην λειτουργία αυτή, χρησιμοποιούνται μια λίστα ( ArrayList ) για την προσωρινή αποθήκευση των σημείων που έχουν ανακτηθεί έως τώρα καθώς και μία στοίβα για την αποθήκευση των κόμβων που απομένουν να επισκεφτούν. Στο τέλος της διαδικασίας η λίστα μετατρέπεται σε πίνακα για ευκολότερη μεταχείριση.

Στο ερώτημα αυτό εφαρμόζεται η πρακτική του κλαδέματος κόμβων με βάση την μικρότερη απόσταση που απέχουν από το σημείο αναζήτησης.

Ερώτημα Πλησιέστερων Γειτόνων:

Στην λειτουργία αυτή, χρησιμοποιείται μια ταξινομημένη λίστα ( ArrayList ) για την αποθήκευση των κόμβων που απομένουν να εξεταστούν και οι οποίοι έχουν περάσει τις στρατηγικές κλαδέματος του δένδρου όπως περιγράφονται στο άρθρο του Ρουσσόπουλου.

Ομοίως χρησιμοποιείται πάλι μια ταξινομημένη λίστα ( ArrayList ) για την αποθήκευση των σημείων που έχουν ανακτηθεί και τα οποία δεν έχουν διαγραφεί από τις στρατηγικές κλαδέματος του δένδρου.

Για την χρήση των λιστών δημιουργήθηκαν δύο ψευδό-κλάσεις. Η μία φιλοξενεί τα σημεία που έχουν ανακτηθεί μαζί με την απόσταση τους από το κέντρο της έρευνας. Η άλλη φιλοξενεί τους κόμβους που απομένουν να εξεταστούν καθώς και τις δύο μετρικές αποστάσεις όπως περιγράφονται στο άρθρο του Ρουσσόπουλου.

Ακόμη δημιουργήθηκε και μια κλάση σύγκρισης για την δυνατότητα ταξινόμησης των παραπάνω μέσα στις λίστες.

Στο τέλος της διαδικασίας αυτής μετατρέπεται η λίστα των αποτελεσμάτων σε πίνακα για ευκολότερη μεταχείριση.

Στατιστικά στοιχεία:

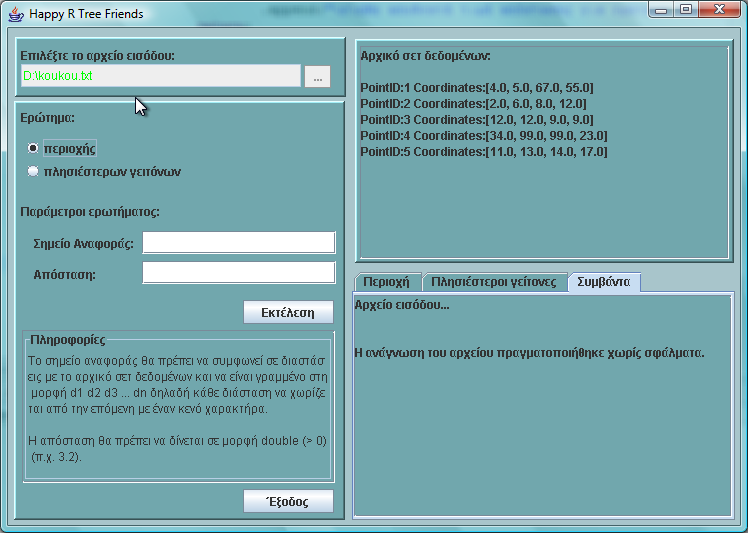
Η δομή αυτή δοκιμάστηκε με την εισαγωγή 3000 σημείων, εκτέλεσης 3000 ερωτημάτων περιοχής και 500 ερωτημάτων εύρεσης των 5-πλησιέστερων γειτόνων. Τα σημεία που χρησιμοποιήθηκαν ακολουθούσαν n-διάστατη κατανομή Gauss με μέση τιμή το 0.0 και τυπική απόκλιση 500. Ακολουθεί ο συγκεντρωτικός πίνακας.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Αριθμός διαστάσεων:** | **2** | **5** | **15** | **30** |
| Εισαγωγή | 2.688 sec | 3.953 sec | 2.672sec | 4.578 sec |
| Ερώτημα περιοχής | 28.203 sec | 45.359 sec | 26.610sec | 9.553 sec |
| Ερώτημα Πλησιέστερων γειτόνων | 21.800 sec | 29.813 sec | 22.125sec | 24.360 sec |

# Γραφικό περιβάλλον

Το GUI αποτελεί ένα κομμάτι της εργασίας που έχει ως στόχο την οπτικοποίηση των λειτουργιών που αναπτύχθηκαν δηλαδή του κτησίματος του RTree από το πρόγραμμα και των ερωτήσεων Περιοχής (Range) και Γειτνίασης (Nearest Neighbours).

Παρακάτω μπορεί κανείς να δει το παράθυρο στην πλήρη του μορφή:



**4**

**3**

**2**

**1**

Το παράθυρο στην πλήρη του μορφή αποτελείται από τις 4 σημασμένες περιοχές που φαίνονται παραπάνω. Κάθε μια από αυτές αποτελεί ένα panel. Ας δούμε εν συντομία τη λειτουργία κάθε μιας από τις περιοχές αυτές.

Panel επιλογής αρχείου (είσοδος) (1):

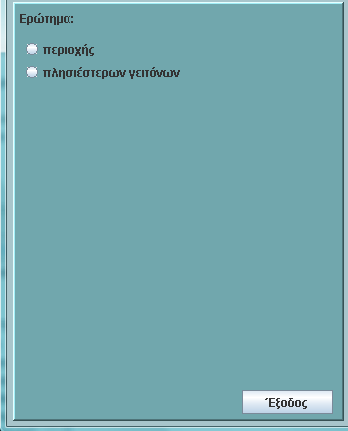
Ο χρήστης καλείται να επιλέξει το αρχείο που περιέχει τα σημεία με βάση τα οποία θα χτισθεί (αρχικοποιηθεί) το RTree. Υπάρχει ένα κουμπί στα δεξιά το οποίο βοηθά τον χρήστη να επιλέξει το επιθυμητό αρχείο. Μόλις αυτός το κάνει και εφόσον το αρχείο δεν εμφανίζει fatal errors (π.χ. δεν υπάρχει ή είναι άδειο) τότε εμφανίζεται η διαδρομή του στο πλαίσιο κειμένου ενώ το κουμπί παγώνει κλειδώνοντας στην επιλογή του χρήστη και μη επιτρέποντας εκ νέου επιλογή. Για να συμβεί κάτι τέτοιο ο χρήστης οφείλει να τρέξει εκ νέου την εφαρμογή.

Panel εμφάνισης αρχικών σημείων (2):

Εδώ εμφανίζεται (εφόσον δεν υπήρξε πρόβλημα με το αρχείο εισόδου) το αρχικό σετ δεδομένων (σημεία) με βάση το οποίο δομείται αρχικά το RTree.

Panel ερωτημάτων (3):

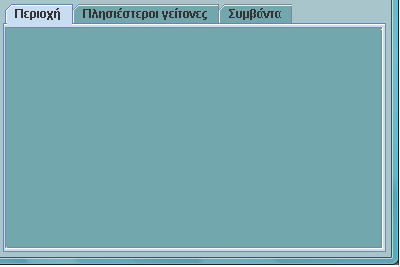
Εδώ ο χρήστης μπορεί να επιλέξει μεταξύ των δύο διαθέσιμων ερωτημάτων (περιοχής και γειτνίασης). Αρχικά το Panel περιέχει μόνον τα κουμπιά επιλογής μεταξύ ερωτημάτων.



Αφού ο χρήστης κάνει την επιλογή του εμφανίζονται τα απαραίτητα πλαίσια για το σχηματισμό του ερωτήματος καθώς και οδηγίες για τις παραδοχές που πρέπει να ακολουθηθούν σε σχέση με τον ορισμό των σημείων αναφοράς καθώς και της επιθυμητής περιοχής ή του αριθμού των γειτόνων.

Panel αποτελεσμάτων & Καταγραφής συμβάντων (4):

Πρόκειται για το πιο σημαντικό τμήμα του γραφικού περιβάλλοντος. Αποτελείται από τρεις καρτέλες.



Περιοχή:

Στην καρτέλα αυτή φιλοξενούνται τα αποτελέσματα της αναζήτησης με βάση ένα σημείο αναφοράς και μια περιοχή ενδιαφέροντος γύρω από αυτό.

Πλησιέστεροι γείτονες:

Αντίστοιχα εδώ φιλοξενούνται τα αποτελέσματα της αναζήτησης με βάση ένα σημείο αναφοράς και έναν αριθμό πλησιέστερων γειτόνων σε αυτό.

Συμβάντα:

Λειτουργεί ως log καταγράφοντας οτιδήποτε συμβαίνει κατά τη λειτουργία του προγράμματος. Ξεκινά με την αρχικοποίηση του RTree ενημερώνοντας τον χρήστη αν το αρχείο διαβάστηκε σωστά, αν υπήρχαν σφάλματα και τι είδους ήταν αυτά ή αν κάτι χρειάζεται να ορισθεί εκ νέου.

Στη συνέχεια παρακολουθεί της κινήσεις του όσον αφορά τα ερωτήματα και εμφανίζει μηνύματα λάθους οδηγώντας το χρήστη να δομήσει σωστά τα ερωτήματά του.

Αξίζει να σημειωθεί ότι στις δύο πρώτες καρτέλες εμφανίζεται ένα πλήθος σημείων ή μήνυμα αποτυχίας αν δεν ήταν δυνατή η εύρεση σημείων που να επαληθεύουν το ερώτημα.

# Παράρτημα – Javadoc

**Package rtree**

|  |  |
| --- | --- |
| **Class Summary** | |
| [**Rtree**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html) | A modified implementation of the R-Tree data structure. |

## rtree Class Rtree

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by **rtree.Rtree**

public class **Rtree**

extends [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

A modified implementation of the R-Tree data structure. It supports Points (instead of Rectangles as in Guttman's original paper). Points can be inserted at any time. It also supports a Range Query based on a Point and a radius and k-Nearest Neighbours Query.  
  
The Range Query is based on minDistance as described in Roussopoulos paper and the methodology as described in Guttman's paper.

The k-Nearest Neighbours Query is based on Roussopoulos paper,with a minor efficiency change on calculating MinMaxDistance

|  |  |
| --- | --- |
| **Nested Class Summary** | |
| private  class | [Rtree.NNeighbourEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.NNeighbourEntry.html)            This is a wrapper of an entry for the Nearest Neighbour Query. |
| private  class | [Rtree.NNeighbourEntryComparator](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.NNeighbourEntryComparator.html)            A comparator for the [Rtree.NNeighbourEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.NNeighbourEntry.html). |
| private  class | [Rtree.NNeighbourRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.NNeighbourRectangle.html)            This is a wrapper of an Node/Rectangle for the Nearest Neighbour Query. |

|  |  |
| --- | --- |
| **Field Summary** | |
| private  int | [height](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#height)            The height of the tree. |
| private  [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[Integer](http://java.sun.com/javase/6/docs/api/java/lang/Integer.html?is-external=true)> | [parentEntryIndex](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#parentEntryIndex)            A stack that specifies the index that was followed, in the parent's entries array, in order to reach this node. |
| private  [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html)> | [parents](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#parents)            A stack that is used when the tree is traversed and stores the parents of each visited node. |
| private  long | [pointCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#pointCount)            The number of points currently in the tree. |
| private  [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) | [root](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#root)            The root of the tree. |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [Rtree](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#Rtree%28%29)()            Constructs a R-Tree and initializes all the data structures in order for the tree to function. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| private  [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) | [adjustTree](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#adjustTree%28rtree.util.Node,%20rtree.util.Node%29)([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) leaf, [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newLeaf)            Ascend from a leaf node L to the root, adjusting MinimumBoundaryRectangles and propagating node splits as necessary.Used by [insert(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#insert%28rtree.util.Point%29) ( Also taken from Guttman's paper). |
| private  [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) | [chooseLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#chooseLeaf%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)            Chooses a leaf to add the rectangle to.Used by [insert(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#insert%28rtree.util.Point%29) ( Also taken from Guttman's paper). |
| void | [insert](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#insert%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)            Inserts a point to the Tree-Structure. |
| private  double | [minmaxDistance](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#minmaxDistance%28rtree.util.Rectangle,%20rtree.util.Point%29)([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) minimumBoundryRectangle, [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre)            Calculates the MinMaxDistance as described to Roussopoulos paper. |
| [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] | [nearestNeighbours](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#nearestNeighbours%28rtree.util.Point,%20int%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre, int noNeighbours)            Implements a nearest Neighbours search from a point of origin, according to Roussopoulos paper. |
| [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] | [rangeQuery](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#rangeQuery%28rtree.util.Point,%20double%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre, double distance)            Calculates a range query.Given the point of origin and the distance finds all the points within this area. |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [finalize](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#finalize%28%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### root

private [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) **root**

The root of the tree.  
It is an InnerNode unless the tree is consisted only by this node;  
In that case it's LeafNode.

### height

private int **height**

The height of the tree.

### pointCount

private long **pointCount**

The number of points currently in the tree.

### parents

private [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html)> **parents**

A stack that is used when the tree is traversed and stores the parents of each visited node.

**See Also:**

[Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)

### parentEntryIndex

private [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[Integer](http://java.sun.com/javase/6/docs/api/java/lang/Integer.html?is-external=true)> **parentEntryIndex**

A stack that specifies the index that was followed, in the parent's entries array, in order to reach this node.

**See Also:**

[parents](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#parents), [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)

|  |
| --- |
| **Constructor Detail** |

### Rtree

public **Rtree**()

Constructs a R-Tree and initializes all the data structures in order for the tree to function.

|  |
| --- |
| **Method Detail** |

### nearestNeighbours

public [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] **nearestNeighbours**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre,

int noNeighbours)

Implements a nearest Neighbours search from a point of origin, according to Roussopoulos paper. It applies all the pruning strategies (both downward and upward).  
The result might have less size than noNeighbours in case the tree contains less Points than noNeighbours.

**Parameters:**

centre - The point of origin.

noNeighbours - The number of the required Neighbours.

**Returns:**

A Point[] containing the results; null in case noNeighbours <= 0.

### minmaxDistance

private double **minmaxDistance**([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) minimumBoundryRectangle,

[Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre)

Calculates the MinMaxDistance as described to Roussopoulos paper.  
It has linear cost (depending on the dimensions of course)

**Parameters:**

minimumBoundryRectangle - The rectangle to calculate the MinMaxDistance from the centre.

centre - The point from which the distance is calculated

**Returns:**

The distance calculated.

### rangeQuery

public [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] **rangeQuery**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) centre,

double distance)

Calculates a range query.Given the point of origin and the distance finds all the points within this area.

**Parameters:**

centre - The point of origin.

distance - The distance.

**Returns:**

A Point[] containing the results. The array might have lenght == 0 in case no results are found.

### insert

public void **insert**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)

Inserts a point to the Tree-Structure.  
It invokes [chooseLeaf(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#chooseLeaf%28rtree.util.Point%29) to find the appropriate leaf to insert to. Checks whether the leaf is full and in that case invokes [LeafNode.splitNode(Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#splitNode%28rtree.util.Entry%29). Finally it propagates the changes upwards using [adjustTree(Node, Node)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#adjustTree%28rtree.util.Node,%20rtree.util.Node%29) and if a root split has occurred, it creates a new root.(Taken from Guttman's paper)

**Parameters:**

point - The point to be inserted

**See Also:**

[chooseLeaf(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#chooseLeaf%28rtree.util.Point%29), [adjustTree(Node, Node)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#adjustTree%28rtree.util.Node,%20rtree.util.Node%29), [LeafNode.splitNode(Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#splitNode%28rtree.util.Entry%29)

### adjustTree

private [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) **adjustTree**([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) leaf,

[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newLeaf)

Ascend from a leaf node L to the root, adjusting MinimumBoundaryRectangles and propagating node splits as necessary.Used by [insert(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#insert%28rtree.util.Point%29)  
( Also taken from Guttman's paper).

**Parameters:**

leaf - The leaf that the point was insert.

newLeaf - The new leaf if a split has occurred;null otherwise.

**Returns:**

The new root if a root split occurred;else null.

### chooseLeaf

private [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) **chooseLeaf**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)

Chooses a leaf to add the rectangle to.Used by [insert(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Rtree.html#insert%28rtree.util.Point%29)  
( Also taken from Guttman's paper).

**Package rtree.util**

|  |  |
| --- | --- |
| **Class Summary** | |
| [**DiskHandler**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\DiskHandler.html) | This class implements a handler for the static data inside our disk. |
| [**Entry**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) | This class represents an entry of our R-tree implementation. |
| [**InnerNode**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html) | The representation of an InnerNode inside an R-Tree. |
| [**LeafNode**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) | The representation of an LeafNode inside an R-Tree. |
| [**Node**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) | This class represents the Nodes of the tree. |
| [**Point**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) |  |
| [**Rectangle**](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) | This is a basic representation of a multidimensional rectangle. |

## rtree.util Class DiskHandler

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by **rtree.util.DiskHandler**

public abstract class **DiskHandler**

extends [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

This class implements a handler for the static data inside our disk. It retrieves and stores the contents of the R-tree into the appropriate files.

**Author:**

Sskalist

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [DiskHandler](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\DiskHandler.html#DiskHandler%28%29)() |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| static [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] | [loadFromDisk](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\DiskHandler.html#loadFromDisk%28long,%20int,%20int%29)(long pageNumber, int entryCount, int maxEntries)            It loads a page from the disk and returns it to a [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) array. |
| static void | [writeToDisk](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\DiskHandler.html#writeToDisk%28rtree.util.Point%5B%5D,%20int,%20long%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] entries, int entryCount, long pageNumber)            It writes a page to a file on the disk. |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [finalize](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#finalize%28%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Constructor Detail** |

### DiskHandler

public **DiskHandler**()

|  |
| --- |
| **Method Detail** |

### loadFromDisk

public static [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] **loadFromDisk**(long pageNumber,

int entryCount,

int maxEntries)

It loads a page from the disk and returns it to a [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) array.

**Parameters:**

pageNumber - The number of the page-target.

maxEntries - The maximum entries.

**Returns:**

An maxEntries-lengthed array containing the points loaded.  
it has from start to entryCount -1 the points loaded and until the end null.

### writeToDisk

public static void **writeToDisk**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)[] entries,

int entryCount,

long pageNumber)

It writes a page to a file on the disk.

**Parameters:**

entries - The entries which are about to be written to the file.

entryCount - The number of entries in the Page/Node

pageNumber - The number that shows the index of the page.

## rtree.util Class Entry

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by **rtree.util.Entry**

**Direct Known Subclasses:**

[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html), [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html)

public abstract class **Entry**

extends [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

This class represents an entry of our R-tree implementation. It can be either [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) or [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Author:**

Sskalist

|  |  |
| --- | --- |
| **Field Summary** | |
| protected  long | [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID)            The id number of the entry. |
| private static long | [maxUsedEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#maxUsedEntryId)            The maximum Id that has been used |
| private static long | [noEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#noEntries)            The entry count. |
| private static [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[Long](http://java.sun.com/javase/6/docs/api/java/lang/Long.html?is-external=true)> | [unusedEntriesStack](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#unusedEntriesStack)            A stack that holds all the unused Ids. |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#Entry%28%29)()            Default constructor. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| protected  void | [finalize](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#finalize%28%29)()            Finalizes the Entry.Inserts its [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) to the stack so it can be used later on. |
| long | [getEntryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getEntryID%28%29)()            Gets the id of the entry. |
| static long | [getNextEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29)()            Gets the next available Id. |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### entryID

protected long **entryID**

The id number of the entry.

### unusedEntriesStack

private static [Stack](http://java.sun.com/javase/6/docs/api/java/util/Stack.html?is-external=true)<[Long](http://java.sun.com/javase/6/docs/api/java/lang/Long.html?is-external=true)> **unusedEntriesStack**

A stack that holds all the unused Ids.

### noEntries

private static long **noEntries**

The entry count.

### maxUsedEntryId

private static long **maxUsedEntryId**

The maximum Id that has been used

|  |
| --- |
| **Constructor Detail** |

### Entry

public **Entry**()

Default constructor. Assigns [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) by calling [getNextEntryId()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29)

|  |
| --- |
| **Method Detail** |

### getNextEntryId

public static long **getNextEntryId**()

Gets the next available Id.

**Returns:**

++[maxUsedEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#maxUsedEntryId) if the [unusedEntriesStack](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#unusedEntriesStack) is empty else the top of the stack.

### getEntryID

public long **getEntryID**()

Gets the id of the entry.

**Returns:**

the entryID

### finalize

protected void **finalize**()

throws [Throwable](http://java.sun.com/javase/6/docs/api/java/lang/Throwable.html?is-external=true)

Finalizes the Entry.Inserts its [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) to the stack so it can be used later on.

**Overrides:**

[finalize](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#finalize%28%29) in class [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

**Throws:**

[Throwable](http://java.sun.com/javase/6/docs/api/java/lang/Throwable.html?is-external=true)

**See Also:**

[Object.finalize()](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#finalize%28%29)

## rtree.util Class InnerNode

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by [rtree.util.Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

extended by [rtree.util.Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

extended by **rtree.util.InnerNode**

public class **InnerNode**

extends [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

The representation of an InnerNode inside an R-Tree.  
It has InnerNodes or LeafNodes as entries. It supports all the basic functionality of a Node. Some of it is implemented in [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)'s level.  
Among these Add/Remove an entry & split the node.

**Author:**

Sskalist

|  |
| --- |
| **Field Summary** |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) |
| [entries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entries), [entryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entryCount), [MaxEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MaxEntries), [MinEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MinEntries), [minimumBoundaryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#minimumBoundaryRectangle) |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#InnerNode%28%29)()            Constructor. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [addEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#addEntry%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)            Adds the specified entry. |
| [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) | [getEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#getEntry%28int%29)(int index)            Gets the specified entry. |
| boolean | [isLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#isLeaf%28%29)()            Because it's not a leaf it always returns false. |
| [InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html) | [splitNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#splitNode%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)            Splits this innerNode. |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) |
| [getEntryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntryCount%28%29), [getMinimumBoundryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getMinimumBoundryRectangle%28%29), [isFull](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isFull%28%29), [performSplit](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#performSplit%28rtree.util.Node,%20rtree.util.Node,%20rtree.util.Entry%29), [removeEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeEntry%28int%29), [removeLastEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeLastEntry%28%29) |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [finalize](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#finalize%28%29), [getEntryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getEntryID%28%29), [getNextEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29) |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Constructor Detail** |

### InnerNode

public **InnerNode**()

Constructor. Calls the [Node.Node()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#Node%28%29) and initializes the [Node.entries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entries) array.

|  |
| --- |
| **Method Detail** |

### addEntry

public void **addEntry**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)

Adds the specified entry.

**Specified by:**

[addEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#addEntry%28rtree.util.Entry%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

toBeAdded - The entry to add.

**See Also:**

[Node.addEntry(Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#addEntry%28rtree.util.Entry%29)

### getEntry

public [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) **getEntry**(int index)

Gets the specified entry.

**Overrides:**

[getEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntry%28int%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

index - The specified index

**Returns:**

The specified Node-entry.

**See Also:**

[Node.getEntry(int)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntry%28int%29)

### splitNode

public [InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html) **splitNode**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)

Splits this innerNode.

**Specified by:**

[splitNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#splitNode%28rtree.util.Entry%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

splitGuilty - The entry that caused the split.

**Returns:**

The new InnerNode created.

**See Also:**

[Node.splitNode(rtree.util.Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#splitNode%28rtree.util.Entry%29), [Node.performSplit(Node, Node, Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#performSplit%28rtree.util.Node,%20rtree.util.Node,%20rtree.util.Entry%29)

### isLeaf

public boolean **isLeaf**()

Because it's not a leaf it always returns false.

**Specified by:**

[isLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isLeaf%28%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Returns:**

false.

**See Also:**

[Node.isLeaf()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isLeaf%28%29)

## rtree.util Class LeafNode

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by [rtree.util.Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

extended by [rtree.util.Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

extended by **rtree.util.LeafNode**

public class **LeafNode**

extends [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

The representation of an LeafNode inside an R-Tree.  
It has Points as entries. It supports all the basic functionality of a Node. Some of it is implemented in [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)'s level.  
Among these Add/Remove an entry & split the node.

**Author:**

Sskalist

|  |  |
| --- | --- |
| **Field Summary** | |
| private  boolean | [pageChanged](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#pageChanged)            States whether the page has changed or not. |
| private static long | [pageCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#pageCount)            Keeps track of how many pages have been created. |
| private  boolean | [pageLoaded](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#pageLoaded)            States whether the page is loaded or not. |
| private  long | [pageNumber](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#pageNumber)            The number of the page assigned to the this LeafNode. |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) |
| [entries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entries), [entryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entryCount), [MaxEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MaxEntries), [MinEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MinEntries), [minimumBoundaryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#minimumBoundaryRectangle) |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#LeafNode%28%29)()            Constructor. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [addEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#addEntry%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)            Adds an entry to the leaf. |
| [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) | [getEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#getEntry%28int%29)(int index)            Gets the entry at the specified index. |
| boolean | [isLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#isLeaf%28%29)()            Because it's a leaf it always returns true. |
| private  boolean | [isPageLoaded](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#isPageLoaded%28%29)()            Check whether the page is loaded or not. |
| void | [loadPage](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#loadPage%28%29)()            Loads page of this leaf. |
| [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) | [removeLastEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#removeLastEntry%28%29)()            Removes the last entry. |
| [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) | [splitNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#splitNode%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)            Splits a leafNode. |
| void | [unloadPage](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#unloadPage%28%29)()            Unloads the page. |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) |
| [getEntryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntryCount%28%29), [getMinimumBoundryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getMinimumBoundryRectangle%28%29), [isFull](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isFull%28%29), [performSplit](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#performSplit%28rtree.util.Node,%20rtree.util.Node,%20rtree.util.Entry%29), [removeEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeEntry%28int%29) |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [finalize](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#finalize%28%29), [getEntryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getEntryID%28%29), [getNextEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29) |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### pageCount

private static long **pageCount**

Keeps track of how many pages have been created.

### pageNumber

private long **pageNumber**

The number of the page assigned to the this LeafNode.

### pageLoaded

private boolean **pageLoaded**

States whether the page is loaded or not.

### pageChanged

private boolean **pageChanged**

States whether the page has changed or not.

|  |
| --- |
| **Constructor Detail** |

### LeafNode

public **LeafNode**()

Constructor. Calls the [Node.Node()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#Node%28%29) and initializes the [Node.entries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entries) array. Also assigns a page number to the leaf and allocates the necessary space to the disk.

|  |
| --- |
| **Method Detail** |

### addEntry

public void **addEntry**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)

Adds an entry to the leaf.  
The page must be loaded! Otherwise a runtime error is thrown.

**Specified by:**

[addEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#addEntry%28rtree.util.Entry%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

toBeAdded - The entry to add.

**See Also:**

[Node.addEntry(Entry)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#addEntry%28rtree.util.Entry%29)

### splitNode

public [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) **splitNode**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)

Splits a leafNode. Alters the current leaf and return the newly created.

**Specified by:**

[splitNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#splitNode%28rtree.util.Entry%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

splitGuilty - The entry that causes the leaf to split.

**Returns:**

The leafNode created.

### removeLastEntry

public [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) **removeLastEntry**()

Removes the last entry.  
The page must be loaded! Otherwise a runtime error is thrown.

**Overrides:**

[removeLastEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeLastEntry%28%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Returns:**

The removed entry.

**See Also:**

[Node.removeLastEntry()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeLastEntry%28%29)

### loadPage

public void **loadPage**()

Loads page of this leaf.

### unloadPage

public void **unloadPage**()

Unloads the page. If it hasn't changed or it isn't loaded it has no effect.

### isPageLoaded

private boolean **isPageLoaded**()

Check whether the page is loaded or not.

**Returns:**

[pageLoaded](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#pageLoaded)

### getEntry

public [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) **getEntry**(int index)

Gets the entry at the specified index.  
The page must be loaded! Otherwise a runtime error is thrown.

**Overrides:**

[getEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntry%28int%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Parameters:**

index - The index of the required entry.

**Returns:**

the specified entry.

**See Also:**

[Node.getEntry(int)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntry%28int%29)

### isLeaf

public boolean **isLeaf**()

Because it's a leaf it always returns true.

**Specified by:**

[isLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isLeaf%28%29) in class [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html)

**Returns:**

true.

**See Also:**

[Node.isLeaf()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isLeaf%28%29)

## rtree.util Class Node

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by [rtree.util.Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

extended by **rtree.util.Node**

**Direct Known Subclasses:**

[InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html), [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html)

public abstract class **Node**

extends [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

This class represents the Nodes of the tree.  
They can be either [InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html) or [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html).  
It provides basic functionality that is common between the above classes.

**Author:**

Sskalist

**See Also:**

[InnerNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html), [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html)

|  |  |
| --- | --- |
| **Field Summary** | |
| protected  [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)[] | [entries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entries)            The entries of the Node... |
| protected  int | [entryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entryCount)            The number of entries currently in the Node |
| static int | [MaxEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MaxEntries)            The maximum entries that a Node can host. |
| static int | [MinEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MinEntries)            The minimum entries that a Node can host. |
| protected  [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) | [minimumBoundaryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#minimumBoundaryRectangle)            The minimum boundary rectangle for the entries of the Node |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#Node%28%29)()            The default constructor. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| abstract  void | [addEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#addEntry%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)            Adds an entry. |
| [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) | [getEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntry%28int%29)(int index)            The entry at index |
| int | [getEntryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getEntryCount%28%29)()            Gets the number of entries. |
| [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) | [getMinimumBoundryRectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#getMinimumBoundryRectangle%28%29)()            Gets The minimumBoundryRectangle. |
| boolean | [isFull](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isFull%28%29)()            Checks if the Node is Full |
| abstract  boolean | [isLeaf](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#isLeaf%28%29)()            Checks whether the Node is a leaf or not. |
| protected static void | [performSplit](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#performSplit%28rtree.util.Node,%20rtree.util.Node,%20rtree.util.Entry%29)([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode, [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode, [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)            Performs the split. |
| private static void | [pickNext](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#pickNext%28rtree.util.Node,%20rtree.util.Node,%20int%29)([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode, [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode, int assignedToOriginal)            Picks and assigns the next entry to be assigned to a group during a node split. |
| private static void | [pickSeeds](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#pickSeeds%28rtree.util.Node,%20rtree.util.Node,%20rtree.util.Entry%29)([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode, [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode, [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)            Guttman's pick Seeds. |
| [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) | [removeEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeEntry%28int%29)(int index)            Removes the entry at index. |
| [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) | [removeLastEntry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#removeLastEntry%28%29)()            Removes the last entry. |
| abstract  [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) | [splitNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#splitNode%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)            Splits the node. |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [finalize](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#finalize%28%29), [getEntryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getEntryID%28%29), [getNextEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29) |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### entries

protected [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)[] **entries**

The entries of the Node... An InnerNode has Nodes as entries; a LeafNode has Points as entries;

### entryCount

protected int **entryCount**

The number of entries currently in the Node

### minimumBoundaryRectangle

protected [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) **minimumBoundaryRectangle**

The minimum boundary rectangle for the entries of the Node

### MaxEntries

public static final int **MaxEntries**

The maximum entries that a Node can host. It is calculated based on [Main.PAGE\_SIZE](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\Main.html#PAGE_SIZE) and the Point#DIMENSIONS.

### MinEntries

public static final int **MinEntries**

The minimum entries that a Node can host. Equals to [MaxEntries](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#MaxEntries)/2

|  |
| --- |
| **Constructor Detail** |

### Node

public **Node**()

The default constructor. It calls the above constructor to assign the EntryId which is inherited and initializes the table of Entries with length equal to MaxEntries

|  |
| --- |
| **Method Detail** |

### addEntry

public abstract void **addEntry**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) toBeAdded)

Adds an entry. The way is handled lower on the class hierarchy.

**Parameters:**

toBeAdded - The entry to add.

### splitNode

public abstract [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) **splitNode**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)

Splits the node. Alters the current and returns the new one.The way is handled lower on the class hierarchy.

**Parameters:**

splitGuilty - The entry that caused the node to split.

**Returns:**

The new Node

### isLeaf

public abstract boolean **isLeaf**()

Checks whether the Node is a leaf or not.

**Returns:**

True if [LeafNode](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html) else false.

**See Also:**

[LeafNode.isLeaf()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\LeafNode.html#isLeaf%28%29), [InnerNode.isLeaf()](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\InnerNode.html#isLeaf%28%29)

### performSplit

protected static void **performSplit**([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode,

[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode,

[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)

Performs the split. Removes entries from the orginalNode and assigns them to the newNode  
The split is performed according to the original Guttman's algorithm with linear cost.

**Parameters:**

originalNode - The Node that is full.

newNode - The new Node that will host some of the originalNode entries.

splitGuilty - The entry that caused originalNode to split.

### pickSeeds

private static void **pickSeeds**([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode,

[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode,

[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) splitGuilty)

Guttman's pick Seeds. It picks two entries (one for each node) and assigns them.

**Parameters:**

originalNode - The Node that is full.

newNode - The new Node that will host some of the originalNode entries.

splitGuilty - The entry that caused originalNode to split.

### pickNext

private static void **pickNext**([Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) originalNode,

[Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) newNode,

int assignedToOriginal)

Picks and assigns the next entry to be assigned to a group during a node split.

### removeEntry

public [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) **removeEntry**(int index)

Removes the entry at index.

**Parameters:**

index - The specified index

**Returns:**

If index <= [entryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entryCount) returns the removed Entry; otherwise null.

### removeLastEntry

public [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) **removeLastEntry**()

Removes the last entry.

**Returns:**

The removed entry.

### getEntry

public [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) **getEntry**(int index)

The entry at index

**Parameters:**

index - The specified index

**Returns:**

If is inside of bounds, the specified entry else throws RuntimeException

### getMinimumBoundryRectangle

public [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) **getMinimumBoundryRectangle**()

Gets The minimumBoundryRectangle.

**Returns:**

the minimumBoundryRectangle

### isFull

public boolean **isFull**()

Checks if the Node is Full

**Returns:**

true in case is full; else false.

### getEntryCount

public int **getEntryCount**()

Gets the number of entries.

**Returns:**

the [entryCount](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html#entryCount).

## rtree.util Class Point

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by [rtree.util.Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

extended by **rtree.util.Point**

public class **Point**

extends [Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html)

**Author:**

Sskalist

|  |  |
| --- | --- |
| **Field Summary** | |
| private  double[] | [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates)            An array containing the coordinate for each of the #DIMENSIONS |
| private  long | [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID)            The pointID. |

|  |
| --- |
| **Fields inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID) |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#Point%28double%5B%5D%29)(double[] coordinates)            Constructs a point with the given coordinates having [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID) equal to [Entry.entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID). |  |
| [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#Point%28long,%20double%5B%5D%29)(long pointID, double[] coordinates)            Constructs a point with the given coordinates and the given pointID. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| double | [distance](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#distance%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) from)            Calculates the Euclidian Distance between this [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) and the [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) passes as parameter. |
| double | [getCoordinate](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#getCoordinate%28int%29)(int index)            Gets the coordinate at the specified index. |
| double[] | [getCoordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#getCoordinates%28%29)()            Clones the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates) |
| double[] | [getCoordinatesNoClone](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#getCoordinatesNoClone%28%29)()            Gets the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates) array without cloning it. |
| long | [getPointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#getPointID%28%29)()            Gets the id of the point. |
| [String](http://java.sun.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [toString](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#toString%28%29)()            Creates a string containing the [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID) and using [Arrays.toString(double[])](http://java.sun.com/javase/6/docs/api/java/util/Arrays.html?is-external=true#toString%28double%5B%5D%29) to transform the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates) array.The String representation is in the form:  PointID:10 Coordinates:[1.2,3.4] |

|  |
| --- |
| **Methods inherited from class rtree.util.**[Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) |
| [finalize](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#finalize%28%29), [getEntryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getEntryID%28%29), [getNextEntryId](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#getNextEntryId%28%29) |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### coordinates

private double[] **coordinates**

An array containing the coordinate for each of the #DIMENSIONS

### pointID

private long **pointID**

The pointID. If not specified it is assigned the value of [Entry.entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID)

|  |
| --- |
| **Constructor Detail** |

### Point

public **Point**(double[] coordinates)

Constructs a point with the given coordinates having [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID) equal to [Entry.entryID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html#entryID).

**Parameters:**

coordinates - The coordinates of the point.

### Point

public **Point**(long pointID,

double[] coordinates)

Constructs a point with the given coordinates and the given pointID.

**Parameters:**

pointID - The desired pointID.

coordinates - The coordinates of the point.

|  |
| --- |
| **Method Detail** |

### getCoordinates

public double[] **getCoordinates**()

Clones the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates)

**Returns:**

A clone of the coordinates

### getCoordinate

public double **getCoordinate**(int index)

Gets the coordinate at the specified index.

**Parameters:**

index -

**Returns:**

The coordinate at index.

### getCoordinatesNoClone

public double[] **getCoordinatesNoClone**()

Gets the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates) array without cloning it.

**Returns:**

the coordinates

### getPointID

public long **getPointID**()

Gets the id of the point.

**Returns:**

the [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID)

### distance

public double **distance**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) from)

Calculates the Euclidian Distance between this [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) and the [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) passes as parameter.

**Parameters:**

from - The end-point.

**Returns:**

The Euclidian Distance.

### toString

public [String](http://java.sun.com/javase/6/docs/api/java/lang/String.html?is-external=true) **toString**()

Creates a string containing the [pointID](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#pointID) and using [Arrays.toString(double[])](http://java.sun.com/javase/6/docs/api/java/util/Arrays.html?is-external=true#toString%28double%5B%5D%29) to transform the [coordinates](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html#coordinates) array.The String representation is in the form:  
  
PointID:10 Coordinates:[1.2,3.4]

**Overrides:**

[toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29) in class [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

**Returns:**

A String representation of the point.

**See Also:**

[Arrays.toString(double[])](http://java.sun.com/javase/6/docs/api/java/util/Arrays.html?is-external=true#toString%28double%5B%5D%29)

## rtree.util Class Rectangle

[java.lang.Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

extended by **rtree.util.Rectangle**

public class **Rectangle**

extends [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

This is a basic representation of a multidimensional rectangle. It consists of two arrays, min and max, containing the minimum and max values respectively.

**Author:**

Sskalist

|  |  |
| --- | --- |
| **Field Summary** | |
| private  double[] | [max](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#max)            Array containing the maximum value for each dimension |
| private  double[] | [min](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#min)            Array containing the minimum value for each dimension |

|  |  |
| --- | --- |
| **Constructor Summary** | |
| [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#Rectangle%28%29)()            Default Constructor. |  |
| [Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#Rectangle%28double%5B%5D,%20double%5B%5D%29)(double[] min, double[] max)            Constructor. |  |

|  |  |
| --- | --- |
| **Method Summary** | |
| void | [add](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#add%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) entry)            Adds the entry. |
| private  void | [add](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#add%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) toBeAdded)            Computes the union of this rectangle and the passed point, storing the result in this rectangle. |
| private  void | [add](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#add%28rtree.util.Rectangle%29)([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)            Computes the union of this rectangle and the passed rectangle, storing the result in this rectangle. |
| double | [area](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#area%28%29)()            Compute the area of this rectangle. |
| boolean | [contains](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#contains%28rtree.util.Rectangle%29)([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)            Determine whether this rectangle contains the specified rectangle |
| double | [distance](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#distance%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) origin)            Return the distance between this rectangle and the passed point. |
| double | [enlargement](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#enlargement%28rtree.util.Entry%29)([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) entry)            Calculates the how much the rectangle will grow if the entry would be added. |
| double | [enlargement](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#enlargement%28rtree.util.Point%29)([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)            Calculate the area by which this rectangle would be enlarged if the passed point was added to it. |
| double | [enlargement](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#enlargement%28rtree.util.Rectangle%29)([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)            Calculate the area by which this rectangle would be enlarged if added to the passed rectangle. |
| boolean | [equals](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#equals%28java.lang.Object%29)([Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) obgject)            Determine whether this rectangle is equal to a given object. |
| double | [getMaximum](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#getMaximum%28int%29)(int index)            Gets the specified value of the max array |
| double[] | [getMaximumArray](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#getMaximumArray%28%29)()            Gets the max array. |
| double | [getMininimum](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#getMininimum%28int%29)(int index)            Gets the specified value of the min array |
| double[] | [getMininimumArray](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#getMininimumArray%28%29)()            Gets the min array. |
| void | [set](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#set%28double%5B%5D,%20double%5B%5D%29)(double[] min, double[] max)            Sets the size of the rectangle. |
| [String](http://java.sun.com/javase/6/docs/api/java/lang/String.html?is-external=true) | [toString](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#toString%28%29)()            Return a string representation of this rectangle, in the form: [1.2, 3.4],[5.6, 7.8] |

|  |
| --- |
| **Methods inherited from class java.lang.**[Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) |
| [clone](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#clone%28%29), [finalize](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#finalize%28%29), [getClass](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#getClass%28%29), [hashCode](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#hashCode%28%29), [notify](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notify%28%29), [notifyAll](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#notifyAll%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long%29), [wait](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#wait%28long,%20int%29) |

|  |
| --- |
| **Field Detail** |

### min

private double[] **min**

Array containing the minimum value for each dimension

### max

private double[] **max**

Array containing the maximum value for each dimension

|  |
| --- |
| **Constructor Detail** |

### Rectangle

public **Rectangle**()

Default Constructor.  
Initializes the min array with the Double.POSITIVE\_INFINITY value and  
the max array with the Double.NEGATIVE\_INFINITY value.

### Rectangle

public **Rectangle**(double[] min,

double[] max)

Constructor. Constructs an new Rectangle according to the passed arrays using [set(double[], double[])](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#set%28double%5B%5D,%20double%5B%5D%29)

**Parameters:**

min - array containing the minimum value for each dimension; eg { min(x), min(y) }

max - array containing the maximum value for each dimension; eg { max(x), max(y) }

|  |
| --- |
| **Method Detail** |

### set

public void **set**(double[] min,

double[] max)

Sets the size of the rectangle.

**Parameters:**

min - Array containing the minimum value for each dimension

max - Array containing the maximum value for each dimension

**Throws:**

[RuntimeException](http://java.sun.com/javase/6/docs/api/java/lang/RuntimeException.html?is-external=true) - if the length of either **min** or **max** differs from Rectangle#DIMENSIONS

### distance

public double **distance**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) origin)

Return the distance between this rectangle and the passed point. If the rectangle contains the point, the distance is zero.

**Parameters:**

origin - Point to find the distance to

**Returns:**

distance between this rectangle and the passed point.

### enlargement

public double **enlargement**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) entry)

Calculates the how much the rectangle will grow if the entry would be added. The rectangle is not altered.

**Parameters:**

entry - The entry that would be added

**Returns:**

The enlargement. In case of error returns [Double.POSITIVE\_INFINITY](http://java.sun.com/javase/6/docs/api/java/lang/Double.html?is-external=true#POSITIVE_INFINITY)

**See Also:**

[enlargement(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#enlargement%28rtree.util.Point%29), [enlargement(Rectangle)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#enlargement%28rtree.util.Rectangle%29)

### enlargement

public double **enlargement**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) point)

Calculate the area by which this rectangle would be enlarged if the passed point was added to it. The rectangle is not altered.

**Parameters:**

point - Point to be added to this rectangle, in order to compute the enlargement.

### enlargement

public double **enlargement**([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)

Calculate the area by which this rectangle would be enlarged if added to the passed rectangle. Neither rectangle is altered.

**Parameters:**

rectangle - Rectangle to be added to this rectangle, in order to compute the enlaregment.

### area

public double **area**()

Compute the area of this rectangle.

**Returns:**

The area of this rectangle

### add

public void **add**([Entry](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Entry.html) entry)

Adds the entry. In case the entry is not an instance of [Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) or [Node](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Node.html) throws an Exception.

**Parameters:**

entry - The entry to be added.

**Throws:**

[RuntimeException](http://java.sun.com/javase/6/docs/api/java/lang/RuntimeException.html?is-external=true)

**See Also:**

[add(Point)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#add%28rtree.util.Point%29), [add(Rectangle)](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#add%28rtree.util.Rectangle%29)

### add

private void **add**([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)

Computes the union of this rectangle and the passed rectangle, storing the result in this rectangle.

**Parameters:**

rectangle - Rectangle to add to this rectangle

### add

private void **add**([Point](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Point.html) toBeAdded)

Computes the union of this rectangle and the passed point, storing the result in this rectangle.

**Parameters:**

toBeAdded - Rectangle to add to this rectangle

### getMininimum

public double **getMininimum**(int index)

Gets the specified value of the min array

**Parameters:**

index - The index of the required value.

**Returns:**

The required value

### getMininimumArray

public double[] **getMininimumArray**()

Gets the min array.

**Returns:**

The [min](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#min) array.

### getMaximum

public double **getMaximum**(int index)

Gets the specified value of the max array

**Parameters:**

index - The index of the required value.

**Returns:**

The required value.

### getMaximumArray

public double[] **getMaximumArray**()

Gets the max array.

**Returns:**

The [max](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html#max) array.

### equals

public boolean **equals**([Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true) obgject)

Determine whether this rectangle is equal to a given object. Equality is determined by the bounds of the rectangle.

**Overrides:**

[equals](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#equals%28java.lang.Object%29) in class [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

**Parameters:**

obgject - - The object to compare with this rectangle.

### contains

public boolean **contains**([Rectangle](file:///C:\Users\KrAp\Desktop\doc\doc\rtree\util\Rectangle.html) rectangle)

Determine whether this rectangle contains the specified rectangle

**Parameters:**

rectangle - The rectangle that might be contained by this rectangle

**Returns:**

true if this rectangle contains the passed rectangle; otherwise false

### toString

public [String](http://java.sun.com/javase/6/docs/api/java/lang/String.html?is-external=true) **toString**()

Return a string representation of this rectangle, in the form: [1.2, 3.4],[5.6, 7.8]

**Overrides:**

[toString](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true#toString%28%29) in class [Object](http://java.sun.com/javase/6/docs/api/java/lang/Object.html?is-external=true)

**Returns:**

String String representation of this rectangle.