EVENT: Start with the initial **nqthm** theory.

```
;; Sets; Matt Kaufmann, Dec. 1989, revised March 1990.  The first few events
;; are some basic events about lists.  I'll take the approach that all these
;; basic functions will be disabled once enough algebraic properties have
;; been proved.

;; Theories:

;; (deftheory set-defns
;;  (length properp fix-properp member append subsetp delete
;;          disjoint intersection set-diff setp))
```

DEFINITION:
length (x)
$=$ **if** listp (x) **then** $1 +$ length (cdr (x))
else 0 **endif**

THEOREM: length-nlistp
$(x \simeq \textbf{nil}) \rightarrow (\text{length}\,(x) = 0)$

THEOREM: length-cons
$\text{length}\,(\text{cons}\,(a,\ x)) = (1 + \text{length}\,(x))$

THEOREM: length-append
$\text{length}\,(\text{append}\,(x,\ y)) = (\text{length}\,(x) + \text{length}\,(y))$

EVENT: Disable length.

THEOREM: append-assoc
$\text{append}\,(\text{append}\,(x,\ y),\ z) = \text{append}\,(x,\ \text{append}\,(y,\ z))$

THEOREM: member-cons
$(a \in \text{cons}\,(x,\ l)) = ((a = x) \lor (a \in l))$

THEOREM: member-nlistp
$(l \simeq \textbf{nil}) \rightarrow (a \notin l)$

EVENT: Disable member.

DEFINITION:
subsetp $(x,\ y)$
$=$ **if** $x \simeq \textbf{nil}$ **then** **t**
else $(\text{car}\,(x) \in y) \land \text{subsetp}\,(\text{cdr}\,(x),\ y)$ **endif**

DEFINITION:
subsetp-wit $(x,\ y)$
$=$ **if** $x \simeq$ **nil** **then** **t**
elseif car $(x) \in y$ **then** subsetp-wit (cdr $(x),\ y)$
else car (x) **endif**

THEOREM: subsetp-wit-witnesses
subsetp $(x,\ y)$
$=$ $(\neg\ ((\text{subsetp-wit}\,(x,\ y) \in x) \wedge (\text{subsetp-wit}\,(x,\ y) \notin y)))$

THEOREM: subsetp-wit-witnesses-general-1
$((\text{subsetp-wit}\,(x,\ y) \notin x) \wedge (a \in x)) \rightarrow (a \in y)$

THEOREM: subsetp-wit-witnesses-general-2
$((\text{subsetp-wit}\,(x,\ y) \in y) \wedge (a \in x)) \rightarrow (a \in y)$

EVENT: Disable subsetp-wit-witnesses.

EVENT: Disable subsetp-wit-witnesses-general-1.

EVENT: Disable subsetp-wit-witnesses-general-2.

THEOREM: subsetp-cons-1
$\text{subsetp}\,(\text{cons}\,(a,\ x),\ y) = ((a \in y) \wedge \text{subsetp}\,(x,\ y))$

THEOREM: subsetp-cons-2
$\text{subsetp}\,(l,\ m) \rightarrow \text{subsetp}\,(l,\ \text{cons}\,(a,\ m))$

THEOREM: subsetp-reflexivity
subsetp $(x,\ x)$

THEOREM: cdr-subsetp
subsetp (cdr $(x),\ x)$

THEOREM: member-subsetp
$((x \in y) \wedge \text{subsetp}\,(y,\ z)) \rightarrow (x \in z)$

THEOREM: subsetp-is-transitive
$(\text{subsetp}\,(x,\ y) \wedge \text{subsetp}\,(y,\ z)) \rightarrow \text{subsetp}\,(x,\ z)$

THEOREM: member-append
$(a \in \text{append}\,(x,\ y)) = ((a \in x) \vee (a \in y))$

THEOREM: subsetp-append
$\text{subsetp}\,(\text{append}\,(x,\ y),\ z) = (\text{subsetp}\,(x,\ z) \wedge \text{subsetp}\,(y,\ z))$

THEOREM: subsetp-of-append-sufficiency
$(\text{subsetp}(a,\ b) \lor \text{subsetp}(a,\ c)) \rightarrow \text{subsetp}(a,\ \text{append}(b,\ c))$

THEOREM: subsetp-nlistp
$(x \simeq \textbf{nil}) \rightarrow (\text{subsetp}(x,\ y) \land (\text{subsetp}(y,\ x) = (y \simeq \textbf{nil})))$

THEOREM: subsetp-cons-not-member
$(z \notin x) \rightarrow (\text{subsetp}(x,\ \text{cons}(z,\ v)) = \text{subsetp}(x,\ v))$

EVENT: Disable subsetp.

`;;;;; Other set-theoretic and list-theoretic definitions, and properp observations.`

DEFINITION:
properp (x)
$=$ **if** listp (x) **then** properp (cdr (x))
else $x =$ **nil endif**

DEFINITION:
fix-properp (x)
$=$ **if** listp (x) **then** cons (car (x), fix-properp (cdr (x)))
else nil endif

THEOREM: properp-fix-properp
properp (fix-properp (x))

THEOREM: fix-properp-properp
properp $(x) \rightarrow$ (fix-properp $(x) = x$)

THEOREM: properp-cons
properp (cons $(x,\ y)$) $=$ properp (y)

THEOREM: properp-nlistp
$(x \simeq \textbf{nil}) \rightarrow (\text{properp}(x) = (x = \textbf{nil}))$

THEOREM: fix-properp-cons
fix-properp (cons $(x,\ y)$) $=$ cons (x, fix-properp (y))

THEOREM: fix-properp-nlistp
$(x \simeq \textbf{nil}) \rightarrow$ (fix-properp $(x) =$ **nil**)

THEOREM: properp-append
properp (append $(x,\ y)$) $=$ properp (y)

THEOREM: fix-properp-append
fix-properp (append $(x,\ y)$) $=$ append (x, fix-properp (y))

THEOREM: append-nil
append $(x, \mathbf{nil}) = $ fix-properp (x)

DEFINITION:
delete (x, l)
$=$ **if** listp (l)
then if $x = $ car (l) **then** cdr (l)
else cons (car (l), delete $(x$, cdr $(l)))$ **endif**
else l **endif**

THEOREM: properp-delete
properp (delete $(x, l)) = $ properp (l)

DEFINITION:
disjoint (x, y)
$=$ **if** listp (x) **then** (car $(x) \notin y) \wedge$ disjoint (cdr $(x), y)$
else t endif

DEFINITION:
disjoint-wit (x, y)
$=$ **if** listp (x)
then if car $(x) \in y$ **then** car (x)
else disjoint-wit (cdr $(x), y)$ **endif**
else t endif

THEOREM: disjoint-wit-witnesses
disjoint (x, y)
$=$ $(\neg$ ((disjoint-wit $(x, y) \in x) \wedge$ (disjoint-wit $(x, y) \in y)))$

EVENT: Disable disjoint-wit.

EVENT: Disable disjoint-wit-witnesses.

DEFINITION:
intersection (x, y)
$=$ **if** listp (x)
then if car $(x) \in y$ **then** cons (car (x), intersection (cdr $(x), y)$)
else intersection (cdr $(x), y)$ **endif**
else nil endif

THEOREM: properp-intersection
properp (intersection (x, y))

DEFINITION:
set-diff $(x,\ y)$
$=$ **if** listp (x)
then if car $(x) \in y$ **then** set-diff (cdr $(x),\ y)$
else cons (car (x), set-diff (cdr $(x),\ y)$) **endif**
else nil endif

THEOREM: properp-set-diff
properp (set-diff $(x,\ y)$)

DEFINITION:
setp (x)
$=$ **if** $\neg$ listp (x) **then** $x =$ **nil**
else (car $(x) \notin$ cdr (x)) $\wedge$ setp (cdr (x)) **endif**

THEOREM: setp-implies-properp
setp $(x) \rightarrow$ properp (x)

EVENT: Disable properp.

EVENT: Let us define the theory *set-defns* to consist of the following events: length, properp, fix-properp, member, append, subsetp, delete, disjoint, intersection, set-diff, setp, properp.

```
;; Set theory lemmas
```

THEOREM: delete-cons
delete $(a$, cons $(b,\ x))$
$=$ **if** $a = b$ **then** x
else cons $(b$, delete $(a,\ x))$ **endif**

THEOREM: delete-nlistp
$(x \simeq$ **nil**$) \rightarrow$ (delete $(a,\ x) = x$)

THEOREM: listp-delete
listp (delete $(x,\ l)$)
$=$ **if** listp (l) **then** $(x \neq$ car $(l)) \vee$ listp (cdr (l))
else f endif

THEOREM: delete-non-member
$(x \notin y) \rightarrow$ (delete $(x,\ y) = y$)

THEOREM: delete-delete
delete $(y$, delete $(x,\ z)) =$ delete $(x$, delete $(y,\ z))$

THEOREM: member-delete
$\text{setp}\,(x) \rightarrow ((a \in \text{delete}\,(b,\ x)) = ((a \neq b) \wedge (a \in x)))$

THEOREM: setp-delete
$\text{setp}\,(x) \rightarrow \text{setp}\,(\text{delete}\,(a,\ x))$

EVENT: Disable delete.

THEOREM: disjoint-cons-1
$\text{disjoint}\,(\text{cons}\,(a,\ x),\ y) = ((a \notin y) \wedge \text{disjoint}\,(x,\ y))$

THEOREM: disjoint-cons-2
$\text{disjoint}\,(x,\ \text{cons}\,(a,\ y)) = ((a \notin x) \wedge \text{disjoint}\,(x,\ y))$

THEOREM: disjoint-nlistp
$((x \simeq \mathbf{nil}) \vee (y \simeq \mathbf{nil})) \rightarrow \text{disjoint}\,(x,\ y)$

THEOREM: disjoint-symmetry
$\text{disjoint}\,(x,\ y) = \text{disjoint}\,(y,\ x)$

THEOREM: disjoint-append-right
$\text{disjoint}\,(u,\ \text{append}\,(y,\ z)) = (\text{disjoint}\,(u,\ y) \wedge \text{disjoint}\,(u,\ z))$

THEOREM: disjoint-append-left
$\text{disjoint}\,(\text{append}\,(y,\ z),\ u) = (\text{disjoint}\,(y,\ u) \wedge \text{disjoint}\,(z,\ u))$

THEOREM: disjoint-non-member
$((a \in x) \wedge (a \in y)) \rightarrow (\neg\ \text{disjoint}\,(x,\ y))$

THEOREM: disjoint-subsetp-monotone-second
$(\text{subsetp}\,(y,\ z) \wedge \text{disjoint}\,(x,\ z)) \rightarrow \text{disjoint}\,(x,\ y)$

THEOREM: subsetp-disjoint-2
$(\text{subsetp}\,(x,\ y) \wedge \text{disjoint}\,(y,\ z)) \rightarrow \text{disjoint}\,(z,\ x)$

THEOREM: subsetp-disjoint-1
$(\text{subsetp}\,(x,\ y) \wedge \text{disjoint}\,(y,\ z)) \rightarrow \text{disjoint}\,(x,\ z)$

THEOREM: subsetp-disjoint-3
$(\text{subsetp}\,(x,\ y) \wedge \text{disjoint}\,(z,\ y)) \rightarrow \text{disjoint}\,(x,\ z)$

EVENT: Disable disjoint.

THEOREM: intersection-disjoint
$(\text{intersection}\,(x,\ y) = \mathbf{nil}) = \text{disjoint}\,(x,\ y)$

THEOREM: intersection-nlistp
$((x \simeq \textbf{nil}) \vee (y \simeq \textbf{nil})) \rightarrow (\text{intersection}(x, y) = \textbf{nil})$

THEOREM: member-intersection
$(a \in \text{intersection}(x, y)) = ((a \in x) \wedge (a \in y))$

THEOREM: subsetp-intersection
$\text{subsetp}(x, \text{intersection}(y, z)) = (\text{subsetp}(x, y) \wedge \text{subsetp}(x, z))$

THEOREM: intersection-symmetry
$\text{subsetp}(\text{intersection}(x, y), \text{intersection}(y, x))$

THEOREM: intersection-cons-1
$\text{intersection}(\text{cons}(a, x), y)$
$= \quad \textbf{if } a \in y \textbf{ then } \text{cons}(a, \text{intersection}(x, y))$
$\quad\quad \textbf{else } \text{intersection}(x, y) \textbf{ endif}$

THEOREM: intersection-cons-2
$(a \notin y) \rightarrow (\text{intersection}(y, \text{cons}(a, x)) = \text{intersection}(y, x))$

```
;; The following is needed because DISJOINT-INTERSECTION-COMMUTER,
;; added during polishing, caused the proof of
;; DISJOINT-DOMAIN-CO-RESTRICT (in "alists.events") to fail.
```

THEOREM: intersection-cons-3
$(w \in x)$
$\rightarrow \quad (\text{subsetp}(\text{intersection}(y, \text{cons}(w, z)), x)$
$\quad\quad = \quad \text{subsetp}(\text{intersection}(y, z), x))$

THEOREM: intersection-cons-subsetp
$\text{subsetp}(\text{intersection}(x, y), \text{intersection}(x, \text{cons}(a, y)))$

THEOREM: subsetp-intersection-left-1
$\text{subsetp}(\text{intersection}(x, y), x)$

THEOREM: subsetp-intersection-left-2
$\text{subsetp}(\text{intersection}(x, y), y)$

THEOREM: subsetp-intersection-sufficiency-1
$\text{subsetp}(y, z) \rightarrow \text{subsetp}(\text{intersection}(x, y), z)$

THEOREM: subsetp-intersection-sufficiency-2
$\text{subsetp}(y, z) \rightarrow \text{subsetp}(\text{intersection}(y, x), z)$

THEOREM: intersection-associative
$\text{intersection}(\text{intersection}(x, y), z) = \text{intersection}(x, \text{intersection}(y, z))$

THEOREM: intersection-elimination
$\text{subsetp}(x,\ y) \rightarrow (\text{intersection}(x,\ y) = \text{fix-properp}(x))$

THEOREM: length-intersection
$\text{length}(x) \not< \text{length}(\text{intersection}(x,\ y))$

THEOREM: subsetp-intersection-member
$(\text{subsetp}(\text{intersection}(x,\ y),\ z) \wedge (a \notin z))$
$\rightarrow \quad (((a \in x) \rightarrow (a \notin y)) \wedge ((a \in y) \rightarrow (a \notin x)))$

```
;; The following wasn't needed in the proof about generalization, but it's a nice rule.
```

THEOREM: intersection-append
$\text{intersection}(\text{append}(x,\ y),\ z) = \text{append}(\text{intersection}(x,\ z),\ \text{intersection}(y,\ z))$

```
;; I'd rather just prove that intersection distributes over append on
;; the right but that isn't true.  Congruence relations would probably
;; help a lot with that problem.  In the meantime, I content myself
;; with the following.
```

THEOREM: disjoint-intersection-append
$\text{disjoint}(x,\ \text{intersection}(y,\ \text{append}(z1,\ z2)))$
$= \quad (\text{disjoint}(x,\ \text{intersection}(y,\ z1)) \wedge \text{disjoint}(x,\ \text{intersection}(y,\ z2)))$

```
;; See comment just above DISJOINT-INTERSECTION-APPEND
```

THEOREM: subsetp-intersection-append
$\text{subsetp}(\text{intersection}(u,\ \text{append}(x,\ y)),\ z)$
$= \quad (\text{subsetp}(\text{intersection}(u,\ x),\ z) \wedge \text{subsetp}(\text{intersection}(u,\ y),\ z))$

THEOREM: subsetp-intersection-elimination-lemma
$(\text{subsetp}(y,\ x) \wedge (\neg\ \text{subsetp}(y,\ z)))$
$\rightarrow \quad (\neg\ \text{subsetp}(\text{intersection}(x,\ y),\ z))$

THEOREM: subsetp-intersection-elimination
$\text{subsetp}(y,\ x) \rightarrow (\text{subsetp}(\text{intersection}(x,\ y),\ z) \leftrightarrow \text{subsetp}(y,\ z))$

THEOREM: disjoint-intersection
$\text{disjoint}(\text{intersection}(x,\ y),\ z) = \text{disjoint}(x,\ \text{intersection}(y,\ z))$

THEOREM: subsetp-intersection-monotone-1
$(\text{subsetp}(\text{intersection}(x,\ y),\ z) \wedge \text{subsetp}(x1,\ x))$
$\rightarrow \quad \text{subsetp}(\text{intersection}(x1,\ y),\ z)$

```
;; The lemma SUBSETP-INTERSECTION-MONOTONE-2 below was added during
;; polishing of the final proof in "generalize.events", since the
;; lemma immediately above wasn't enough at that point.  Actually
;; I realized at this point that intersection commutes from the point
;; of view of subsetp:
```

THEOREM: subsetp-intersection-commuter
subsetp (intersection (x, y), z) = subsetp (intersection (y, x), z)

THEOREM: subsetp-intersection-monotone-2
(subsetp (intersection (y, x), z) $\wedge$ subsetp ($x1$, x))
$\rightarrow$ subsetp (intersection ($x1$, y), z)

THEOREM: disjoint-intersection-commuter
disjoint (x, intersection (y, z)) = disjoint (x, intersection (z, y))

THEOREM: disjoint-intersection3
disjoint ($free$, intersection ($vars$, x))
$\rightarrow$ (intersection (x, intersection ($vars$, $free$)) = **nil**)

EVENT: Disable intersection.

THEOREM: member-set-diff
($a \in$ set-diff (y, z)) = (($a \in y$) $\wedge$ ($a \notin z$))

THEOREM: subsetp-set-diff-1
subsetp (set-diff (x, y), x)

THEOREM: disjointp-set-diff
disjoint (set-diff (x, y), y)

THEOREM: subsetp-set-diff-2
subsetp (x, set-diff (y, z)) = (subsetp (x, y) $\wedge$ disjoint (x, z))

THEOREM: set-diff-cons
set-diff (cons (a, x), y)
= **if** $a \in y$ **then** set-diff (x, y)
else cons (a, set-diff (x, y)) **endif**

THEOREM: set-diff-nlistp
($x \simeq$ **nil**) $\rightarrow$ (set-diff (x, y) = **nil**)

```
;; The following was discovered during final polishing, for the
;; proof of MAIN-HYPS-RELIEVED-6-FIRST.
```

THEOREM: disjoint-set-diff-general
disjoint $(x, \text{set-diff}\,(y, z)) = \text{subsetp}\,(\text{intersection}\,(x, y), z)$

```
;; No longer relevant:
;(prove-lemma disjoint-set-diff-subsetp (rewrite)
;  (implies (and (disjoint x (set-diff y z))
; (subsetp z z1))
;    (disjoint x (set-diff y z1)))
;  ((use (disjoint-wit-witnesses (y (set-diff y z1))))
;   (disable member-set-diff set-diff)))

;; Instead of the following I'll prove the corresponding (in light of
;; DISJOINT-SET-DIFF-GENERAL) fact INTERSECTION-X-X about intersection.
;(prove-lemma disjoint-set-diff (rewrite)
;  (disjoint x (set-diff y x)))
```

THEOREM: intersection-subsetp-identity
$(\text{properp}\,(x) \land \text{subsetp}\,(x, y)) \rightarrow (\text{intersection}\,(x, y) = x)$

THEOREM: intersection-x-x
$\text{properp}\,(x) \rightarrow (\text{intersection}\,(x, x) = x)$

THEOREM: subsetp-set-diff-mononone-2
$\text{subsetp}\,(\text{set-diff}\,(x, \text{append}\,(y, z)), \text{set-diff}\,(x, z))$

THEOREM: subsetp-set-diff-monotone-second
$\text{subsetp}\,(\text{set-diff}\,(x, y), \text{set-diff}\,(x, z)) = \text{subsetp}\,(\text{intersection}\,(x, z), y)$

THEOREM: set-diff-nil
$\text{set-diff}\,(x, \textbf{nil}) = \text{fix-properp}\,(x)$

THEOREM: set-diff-cons-non-member-1
$(a \notin x) \rightarrow (\text{set-diff}\,(x, \text{cons}\,(a, y)) = \text{set-diff}\,(x, y))$

THEOREM: length-intersection-set-diff
$\text{length}\,(x) = (\text{length}\,(\text{set-diff}\,(x, y)) + \text{length}\,(\text{intersection}\,(x, y)))$

THEOREM: length-set-diff-opener
$\text{length}\,(\text{set-diff}\,(x, y)) = (\text{length}\,(x) - \text{length}\,(\text{intersection}\,(x, y)))$

THEOREM: listp-set-diff
$\text{listp}\,(\text{set-diff}\,(x, y)) = (\neg\, \text{subsetp}\,(x, y))$

```
;; Here is a messy lemma about disjoint and such
```

THEOREM: disjoint-intersection-set-diff-intersection
disjoint $(x,$ intersection $(y,$ set-diff $(z,$ intersection $(y, x))))$

EVENT: Disable set-diff.

THEOREM: member-fix-properp
$(a \in$ fix-properp $(x)) = (a \in x)$

THEOREM: setp-append
setp (append $(x, y)) =$ (disjoint $(x, y) \land$ setp (fix-properp $(x)) \land$ setp $(y))$

THEOREM: setp-cons
setp (cons $(a, x)) = ((a \notin x) \land$ setp $(x))$

THEOREM: setp-nlistp
$(x \simeq \mathbf{nil}) \rightarrow ($setp $(x) = (x = \mathbf{nil}))$

DEFINITION:
make-set (l)
$=$ **if** $\neg$ listp (l) **then nil**
elseif car $(l) \in$ cdr (l) **then** make-set (cdr (l))
else cons (car (l), make-set (cdr (l))) **endif**

THEOREM: make-set-preserves-member
$(x \in$ make-set $(l)) = (x \in l)$

THEOREM: make-set-preserves-subsetp-1
subsetp (make-set (x), make-set (y)) $=$ subsetp (x, y)

THEOREM: make-set-preserves-subsetp-2
subsetp $(x$, make-set $(y)) =$ subsetp (x, y)

THEOREM: make-set-preserves-subsetp-3
subsetp (make-set $(x), y) =$ subsetp (x, y)

THEOREM: make-set-gives-setp
setp (make-set (x))

THEOREM: make-set-set-diff
make-set (set-diff $(x, y)) =$ set-diff (make-set (x), make-set (y))

THEOREM: set-diff-make-set
set-diff $(x$, make-set $(y)) =$ set-diff (x, y)

THEOREM: listp-make-set
listp (make-set $(x)) =$ listp (x)

EVENT: Disable setp.

```
;;;;;; The following were proved in the course of the final run
;;;;;; through the generalization proof.  There are a couple or
;;;;;; so noted above here, too.
```

THEOREM: set-diff-append
set-diff $(x,\ \text{append}\,(y,\ z)) = \text{set-diff}\,(\text{set-diff}\,(x,\ z),\ y)$

THEOREM: length-set-diff-leq
length $(x) \not< \text{length}\,(\text{set-diff}\,(x,\ y))$

THEOREM: lessp-length
listp $(x) \rightarrow (0 < \text{length}\,(x))$

THEOREM: listp-intersection
listp $(\text{intersection}\,(x,\ y)) = (\neg\ \text{disjoint}\,(x,\ y))$

THEOREM: length-set-diff-lessp
$(\neg\ \text{disjoint}\,(x,\ \mathit{new})) \rightarrow (\text{length}\,(\text{set-diff}\,(x,\ \mathit{new})) < \text{length}\,(x))$

THEOREM: disjoint-implies-empty-intersection
disjoint $(x,\ y) \rightarrow (\text{intersection}\,(x,\ y) = \mathbf{nil})$

```
;; The following lemma DISJOINT-INTERSECTION3-MIDDLE is needed for the
;; proof of ALL-VARS-DISJOINT-OR-SUBSETP-GEN-CLOSURE in
;; generalize.events.  I think I could avoid lemmas like this one
;; INTERSECTION were actually commutative-associative (in which case
;; I'd get rid of disjoint and rely on normalization).
```

THEOREM: disjoint-intersection3-middle
disjoint $(y,\ \text{intersection}\,(x,\ z))$
$\rightarrow \quad (\text{intersection}\,(x,\ \text{intersection}\,(y,\ z)) = \mathbf{nil})$

```
;; Maybe I should redo the notion of disjoint sometime, perhaps using
;; the fact that intersection is commutative and associative when it's
;; equated with nil.
```

THEOREM: disjoint-subsetp-hack
$(\text{disjoint}\,(x,\ \text{intersection}\,(u,\ v)) \land \text{subsetp}\,(w,\ x))$
$\rightarrow \quad \text{disjoint}\,(u,\ \text{intersection}\,(w,\ v))$

THEOREM: subsetp-set-diff-sufficiency
subsetp $(x,\ y) \rightarrow$ subsetp (set-diff $(x,\ z),\ y)$

```
;; The following lemma SETP-INTERSECTION-SUFFICIENCY is needed for
;; MAPPING-RESTRICT from "alists.events", because (I believe)
;; DOMAIN-RESTRICT, which was added during polishing, changed the
;; course of the previous proof.  Similarly for
;; SETP-SET-DIFF-SUFFICIENCY and the lemma MAPPING-CO-RESTRICT.
```

THEOREM: setp-intersection-sufficiency
setp $(x) \rightarrow$ setp (intersection $(x,\ y)$)

THEOREM: setp-set-diff-sufficiency
setp $(x) \rightarrow$ setp (set-diff $(x,\ y)$)

```
;; The definition of FIX-PROPERP was also added in polishing because
;; of a problem with the proof of GEN-CLOSURE-ACCEPT in
;; "generalize.events".  Here are a couple of lemmas about it that
;; might or might not be useful; all other lemmas about it above, and
;; the definition, were added during polishing.
```

EVENT: Disable fix-properp.

THEOREM: subsetp-fix-properp-1
subsetp (fix-properp $(x),\ y) =$ subsetp $(x,\ y)$

THEOREM: subsetp-fix-properp-2
subsetp $(x,$ fix-properp $(y)) =$ subsetp $(x,\ y)$

EVENT: Make the library `"sets"`.

Index